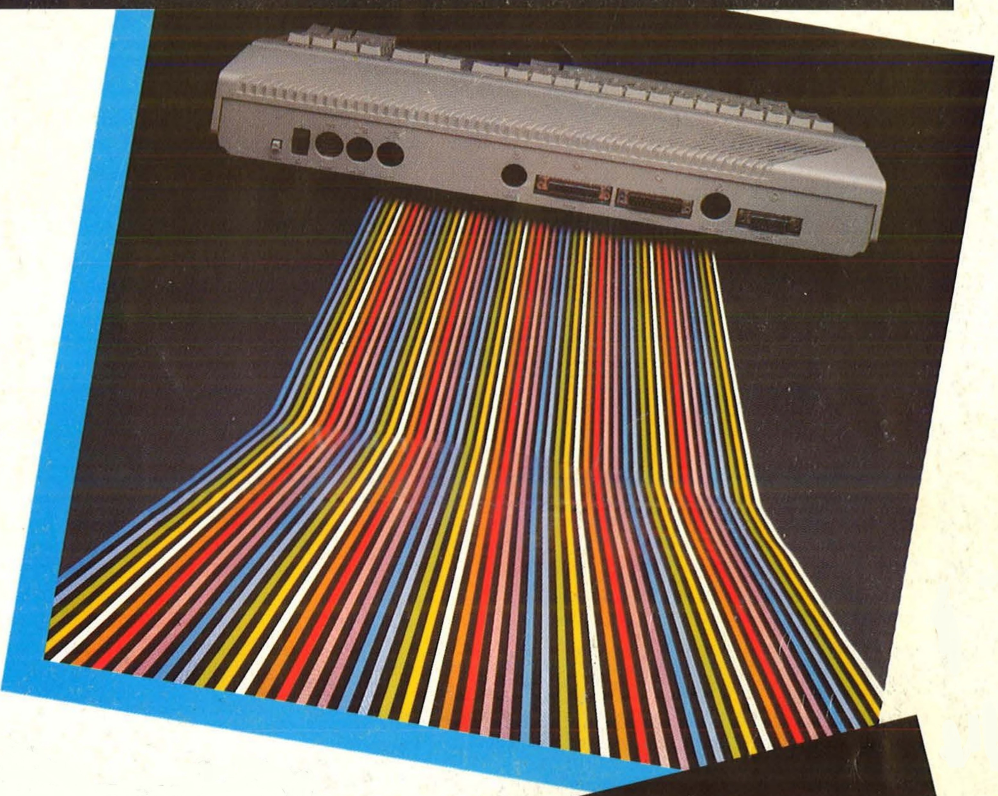
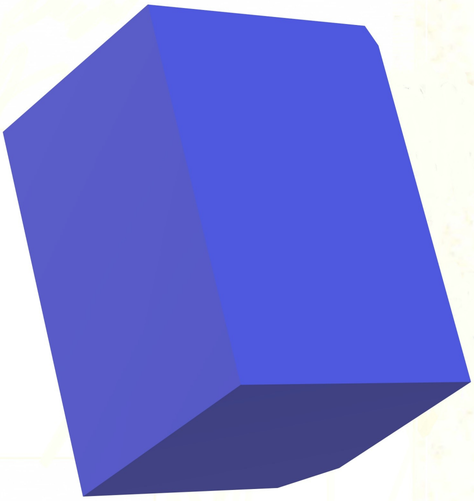
Das Magazin für aktives und kreatives Computern

HC Mein Home-Computer

EXTRA





TI99/4A

COLLECTION

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Homecomputer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC20, Texas Instruments 99/4A und Atari 600 XL

Den Besitzern eines anderen Rechners wird es nicht schwer fallen, ihr Gerät in der Nähe einer der genannten vier Homecomputer anzusiedeln, dem ihr Gerät verwandt ist. Außerdem kann potentiellen Käufern eines Homecomputers die besondere Hervorhebung von Unterschieden zwischen den Rechnern als Entscheidungshilfe dienen.

Nachdem der Homecomputer mit Strom versorgt und über ein Antennenkabel an einen Fernseher angeschlossen ist, wollen wir zunächst wissen, ob mit ihm alles in Ordnung ist und er Eingaben entgegennehmen kann. Jedes Homecomputer-Modell sendet dazu eine typische Meldung aus, mit der es sich vorstellt oder auch angibt, wieviel Speicherplatz es zur Verfügung stellt. Zunächst soll der Homecomputer zeigen, daß er bereit ist, auch bescheidenen Anweisungen Folge zu leisten.

Der Home-Computer zum Taschenrechner degradiert

Das ist eigentlich eine Zumutung für ihn, aber er nimmt es nicht übel.

Er soll die Aufgabe 3 × 8 + 4 ausrechnen und das Ergebnis auf den Bildschirm schreiben. Die Anweisung PRINT vor der Rechenaufgabe hat dies zur Folge.

PRINT zum Schreiben auf den Bildschirm

Das Wort PRINT und die anderen BASIC-Befehle müssen bei manchen Homecomputern Buchstabe für Buchstabe eingegeben werden. Bei anderen kann jeweils ein ganzes Wort (genannt Schlüsselwort) durch Drücken einer (oder von höchstens zwei) Tasten auf den Bildschirm gebracht werden.

Wenn man sich an das Arbeiten mit Schlüsselwörtern gewöhnt hat, geht es natürlich viel schneller (Tab. 1).

Jeder Homecomputer besitzt eine Taste, mit der das Ende einer Eingabe markiert wird.

Zum Lösen der obigen Aufgabe ist also folgende Eingabe nötig:

→ PRINT 3 × 8 + 4
Taste zum Ende der Eingabe

(Wir werden im weiteren Text die Taste zum Ende einer Eingabe nicht mehr erwähnen, weil ihre Benutzung sich schnell einprägt.)

Wenn von Ihnen eine Eingabe in den Computer verlangt wird, so ist dies immer durch einen Pfeil am Anfang der Zeile gekennzeichnet.

Das Ergebnis der kleinen Aufgabe wird je nach Homecomputer-Modell an eine andere Stelle des Bildschirms gedruckt. Das liegt an ihrem unterschiedlichen Bildschirmverhalten. Bisher haben wir ausschließlich den sogenannten Direktbetrieb benutzt. Anweisungen im Direktbetrieb nennt man auch Kommandos. Erst später wenden wir uns dem Programmbetrieb zu (Tab. 2).

Mit dem PRINT-Befehl

können auch kompliziertere Aufgaben berechnet werden. Hier ein Beispiel, in dem die Exponentialfunktion und die Sinusfunktion vorkommen:

PRINT ((3 × 19 + 7) × SIN (0.5) − 3.216) × EXP (−1.2)

Gezieltes Drucken an eine bestimmte Stelle

Der Raum, der auf dem Bildschirm zur Verfügung steht, ist durch die Anzahl der Zeilen und die Anzahl der Spalten unterteilt.

Die Durchnumerierung erfolgt jeweils von links nach rechts und von oben nach unten. Außerdem muß man wissen, ob der Homecomputer die Durchnumerie-

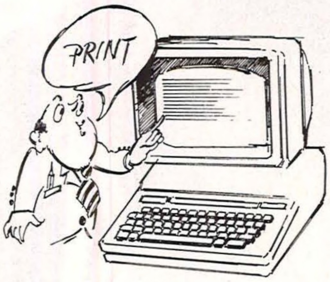
Spectrum	VC20	TI99/4A	Atari 600 XL
Taste /NEW LINE/	Taste /RETURN/	Taste /ENTER/	Taste /RETURN/

Das Ende einer einzelnen Eingabe (Tab. 1)

Spectrum	VC20	TI99/4A	Atari 600 XL
Kommandos erscheinen stets am unteren Bildrand in einer oder mehreren extra reservierten Kommandozeilen. Vor Ausführung des Kommandos wird der Bildschirm gelöscht (auch die Kommandoeingabe).	Der Bildschirm wird von oben nach unten allmählich vollgeschrieben. Danach scrollt der Bildschirminhalt, das heißt er rückt zeilenweise nach oben.	Kommandos erscheinen stets am unteren Bildrand. Der Bildschirminhalt scrollt.	Der Bildschirm wird von oben nach unten allmählich vollgeschrieben. Danach scrollt der Bildschirminhalt.

Bildschirmverhalten bei Eingabe eines Kommandos im Direktbetrieb (Tab. 2)

Scan: S. Höltgen



rung mit 0 oder mit 1 beginnt (Tab. 3).

Wir wollen jetzt das Wort "MUT" in die Zeile 5 beginnend mit der Spalte 10 schreiben. Daß es sich diesmal um einen Text handelt und nicht um eine Zahl, machen wir dem Homecomputer durch Anführungsstriche klar (in der Weise, wie wir es oben schon getan haben) (Tab. 4).

(Beim VC20 erscheint das Zeichen **S** durch Drücken der Taste /CLR HOME/. Hiermit wird erreicht, daß der Cursor nach links oben springt und der Bildschirm gelöscht wird. Das Zeichen **Q** erreicht man durch die Taste /CRSR/. Es bewirkt einen Cursorsprung um eine Zeile nach unten).

Kann man mit Buchstaben rechnen?

Versuchen Sie es! Geben Sie das Kommando

→ PRINT A + B ein!

(Haben Sie C erwartet oder sagen Sie jetzt, daß Sie es ja schon immer gewußt haben, daß man mit Buchstaben nicht rechnen kann?) Die meisten unserer vier Rechner geben als Ergebnis Null an (nur der Spectrum reagiert mit einer Fehlermeldung). Um das zu verstehen, wollen wir etwas weiter ausholen.

Buchstaben können stellvertretend für Zahlen stehen. Ein Buchstabe steht dann jedoch nicht immer für die gleiche Zahl, sondern er kann seinen Wert laufend wechseln: sein Inhalt ist variabel. Man nennt diese Buchstaben deshalb auch Variable und sagt z.B.: die Variable A hat gerade den Wert 5.

In der obigen Aufgabe hatten die Variablen A und B aber noch gar keinen Wert zugewiesen bekommen. Die meisten Rechner setzen in diesem Fall den Wert der Variablen von sich aus auf Null. Deswegen kam auch als Ergebnis der Aufgabe A + B Null heraus (außer beim Spectrum).

Die Zuweisung beispielsweise der Werte 7 und 16 kann die Anweisung LET übernehmen.

LET Zuweisung von Werten zu Variablen

Diese Werte beinhalten die beiden Variablen so lange, bis sie in einem Kommando oder in einem Programm ausdrücklich geändert werden.

Folgende Kommandofolge führt also zum Drucken der Zahl 23:

→ LET A = 7
→ LET B = 16
→ PRINT A + B

Um den Vorrat an Variablennamen zu vergrößern, sind auch Folgen von mehreren Zeichen zugelassen. Namen, die mit einem Buchstaben beginnen, dem sich dann Buchstaben oder Zahlen anschließen, sind auf allen Rechnern erlaubt. Unterschiedlich ist die maximale Länge eines Namens (Tab. 5).

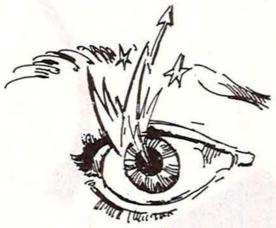
Wir wollen Quadradzahlen berechnen

Eine Möglichkeit ist natürlich, laufend Kommandos einzugeben:

→ PRINT 9 × 9
... PRINT 31 × 31
PRINT 26 × 26 usw.

Der Tippaufwand soll jedoch möglichst gering gehalten werden. Hierzu benutzen wir die Anweisung

PRINT A × A



Sie quadriert den Wert von A und druckt das Ergebnis. Vorher müssen wir der Variablen A jedoch einen Wert zuweisen. Der Homecomputer soll außerdem diese beiden Anweisungen oft wiederholen können, ohne daß wir sie beide jedesmal als Kommandos neu eingeben müssen.

Genau diese Aufgabe übernimmt das folgende (unser erstes) Programm:

→ 5 INPUT A
→ 10 PRINT A × A

Hierzu muß einiges erklärt werden.

Nur dadurch, daß wir Anweisungen Zahlen voranstellen, werden aus Kommandos Programmzeilen. Diese Zahlen sind die Zeilennummern und sie legen fest, in welcher Reihenfolge die Anweisungen später ausgeführt werden sollen. Zeilennummern von 1 bis 9999 sind auf allen vier Rechnern möglich (bei einigen sogar größere); dabei dürfen zwischen den einzelnen Nummern beliebig große Lücken sein.

Die Programmzeilen eines Programmes werden vom Homecomputer in der Reihenfolge ihrer Zeilennummern gespeichert. Sie werden nicht wie die Kommandos sofort ausgeführt und auch nicht danach gleich wieder vergessen. Ein Programm kann später ergänzt oder geändert werden.

Wie Sie schon bemerkt haben, übernimmt in unserem Programm die Zuweisung eines Wertes nicht der Befehl LET, sondern eine neuartige Anweisung:

INPUT A

Wie der Befehl INPUT arbeitet, werden wir sehen, wenn wir unser kleines Programm ausführen.

Der Kurs wird in der nächsten Ausgabe fortgesetzt.

Spectrum	VC20	TI99/4A	Atari 600 XL
22 Zeilen 32 Spalten Durchnumerierung beginnt jeweils mit 0	23 Zeilen 22 Spalten Durchnumerierung beginnt jeweils mit 0	24 Zeilen 32 Spalten Durchnumerierung beginnt jeweils mit 1	24 Zeilen (abweichend in anderem Textmodus) 40 Spalten Durchnumerierung beginnt jeweils mit 0

Anzahl der Zeilen und Spalten und deren Durchnumerierung (Tab. 3)

Spectrum	VC20	TI99/4A	Atari 600 XL
PRINT AT 5, 10; "MUT"	PRINT "**SQQQQQ**"; TAB (10); "MUT"	CALL HCHAR (5, 10, 77, 1) CALL HCHAR (5, 11, 85, 1) CALL HCHAR (5, 12, 84, 1) (nur in einem Programm sinnvoll)	POS. 10,5: PRINT "MUT"

Drucken eines Textes an eine bestimmte Stelle des Bildschirms (Tab. 4)

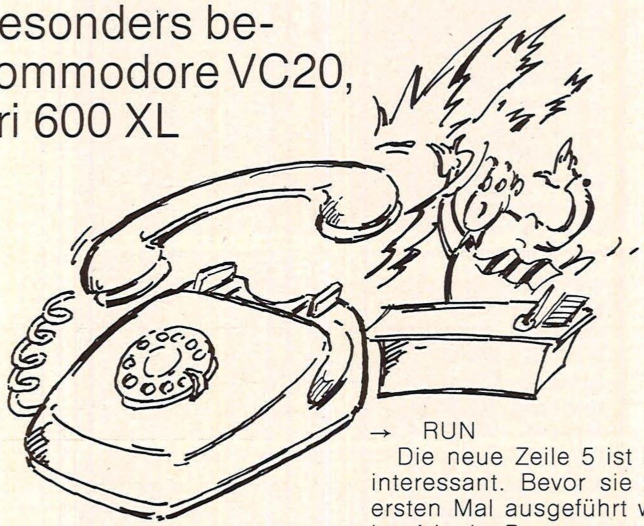
Spectrum	VC20	TI99/4A	Atari 600 XL
Länge unbegrenzt	maximal 2 Zeichen	maximal 15 Zeichen	Länge unbegrenzt

Zugelassene Länge von Zahlvariablen (Tab. 5)

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Homecomputer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC20, Texas Instruments 99/4A und Atari 600 XL



Im ersten Teil des BASIC-Kurses übernahm ein kleines Programm die Aufgabe, Quadratzahlen zu berechnen:

→ 5 INPUT A
→ 10 PRINT A × A

Es wird mit
→ RUN gestartet. Bei der Anweisung INPUT A hält es an und wartet auf die Eingabe einer Zahl.

INPUT Das Programm hält an und erwartet eine Eingabe, die einer Variablen zugewiesen wird

Wenn Sie jetzt z.B.
→ 26 eingeben, wird das Programm fortgesetzt, und es wird die Zahl 676 gedruckt. Damit ist das Programm beendet. Wenn Sie jetzt eine andere Zahl quadrieren wollen, so geben Sie wieder das Kommando
→ RUN ein und danach die neue Zahl.

Der Sprungbefehl

Es stört jetzt noch, daß wir ständig RUN eingeben müssen. Dies wird durch Eingabe der Programmzeile
→ 15 GOTO 5
überflüssig. Damit fügen wir dem aus zwei Zeilen bestehenden Programm eine dritte Zeile hinzu.

Damit Sie sich jederzeit über den momentanen Zustand des Programmes erkundigen können, gibt es die Anweisung
→ LIST. Sie druckt das gesamte Programm.

LIST Drucken des gesamten Programmes

Wenn Sie alle Eingaben in der beschriebenen Weise durchgeführt haben, muß jetzt nach dem Kommando LIST folgender Ausdruck erscheinen:

```
5 INPUT A
10 PRINT A × A
15 GOTO 5
```

Die neue Programmzeile GOTO 5 sorgt dafür, daß zur Anweisung mit der Zeilennummer 5 zurückgesprungen wird. Auf diese Weise wird die Anweisungsfolge des Programmes in einer Schleife ständig wiederholt. Sie brauchen jetzt nur noch einmal
→ RUN und danach die Zahlen, die quadriert werden sollen, einzugeben und es erscheinen nacheinander die gewünschten Quadratzahlen auf dem Bildschirm.

Dieses Programm kommt nicht von selbst zu einem Ende, sondern fordert immer wieder von Neuem eine Zahl an. Sie haben jedoch die Möglichkeit, von sich aus ein Programm jederzeit zu unterbrechen. Das Kommando hierzu ist
→ BREAK.

BREAK unterbricht den Programmablauf (beim VC20 die Taste /RUN STOP/)

Im nächsten Schritt wollen wir uns auch noch die Eingabe der einzelnen Zahlen ersparen. Das Programm soll automatisch alle Quadratzahlen beginnend mit der 1 liefern. Geben Sie dazu ein:

→ 3 LET A = 0
→ 5 LET A = A + 1

Wir haben jetzt zum zweiten Mal eine Anweisung mit der Zeilennummer 5 eingegeben. Dadurch wurde automatisch die Anweisung 5 INPUT A überschrieben; sie befindet sich nicht mehr im Programm. Kontrollieren Sie wieder mit
→ LIST, ob das Programm jetzt wirklich so aussieht, wie Sie es sich vorstellen.

→ RUN

Die neue Zeile 5 ist sehr interessant. Bevor sie zum ersten Mal ausgeführt wird, hat A in der Programmzeile 3 den Wert 0 zugewiesen bekommen. Zeile 5 könnte man also so mißverstehen, daß hier behauptet wird, 0 sei gleich 1. Die Funktion dieser Anweisung ist jedoch anders.

Zunächst wird die Aufgabe, die rechts vom Gleichheitszeichen steht, ausgerechnet:

0 + 1 = 1

Zeile 5 bekommt somit die folgende Bedeutung:

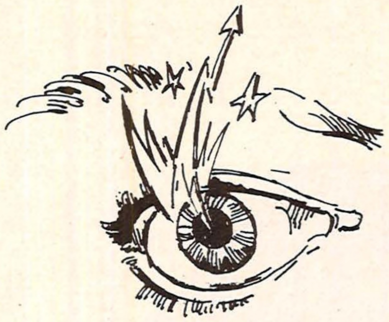
5 LET A = 1

Der Variablen A wird hier also der Wert 1 zugewiesen.

Nachdem die erste Quadratzahl 1 gedruckt wurde, führt das Programm nach 15 GOTO 5 wieder die Anweisung 5 LET A = A + 1 aus. A erhält dadurch auf die gleiche Weise wie oben jetzt den Wert 2 und es wird die Zahl 4 gedruckt. Danach erhält A den Wert 3 usw. Somit wird die Folge der ersten Quadratzahlen fortlaufend gedruckt.

Je nach Homecomputer bricht das Programm beim Erreichen des unteren Bildschirmrandes selbsttätig ab, oder Sie unterbrechen es mit dem Kommando BREAK.

Scan: S. Höltgen



Bedingte Anweisungen

Eleganter ist es, wenn man schon im Programm angeben kann, daß die Rechnung nur bis zu einer bestimmten Quadratzahl ausgeführt werden soll. Dazu darf der Rückwärtssprung in Zeile 15 nur erfolgen, solange A einen bestimmten Grenzwert nicht überschritten hat. Für die ersten 20 Quadratzahlen heißt die neue Programmzeile

→ 15 IF A <= 20 THEN GOTO 5

Es handelt sich hier um eine bedingte Anweisung. Der Sprungbefehl GOTO 5 wird nur ausgeführt, wenn A kleiner oder gleich 20 ist.

Bedingte Anweisungen werden also immer mit einem IF eingeleitet. Es folgt die Bedingung, und danach kommt das Wort THEN und dann die Anweisung. Die Anweisung muß nicht notwendig ein Sprungbefehl sein.

→ RUN

FOR – NEXT – Schleifen

Eine FOR-NEXT-Schleife vereinfacht den Programmaufbau bei einem bedingten Rückwärtssprung.

Da das erforderliche neue Programm sich vom bisherigen wesentlich unterscheidet, wollen wir zunächst das bisherige Programm löschen. Hierzu geben wir das Kommando

→ NEW ein.

NEW löscht ein Programm.

Geben Sie nun das neue Programm ein:

→ 5 FOR A = 1 TO 20
→ 10 PRINT A × A
→ 15 NEXT A

Starten Sie es mit

→ RUN.

Die Anweisung PRINT A × A wird hier durch zwei neuartige Befehle eingerahmt.

FOR A = 1 TO 20 und NEXT A bewirken zusammen, daß alles, was zwischen ihnen steht, zwanzig Mal hintereinander ausgeführt wird, und zwar beim ersten Mal mit dem Wert 1 für A, dann mit dem Wert 2 für A und zuletzt mit dem Wert 20 für A. Dies hat zur Folge, daß die ersten 20 Quadratzahlen gedruckt werden.

A nennt man die Laufvariable der FOR-NEXT-Schleife. Für Namen von Laufvariablen gelten die gleichen Vorschriften wie für Namen von Zahlvariablen (nur beim Spectrum muß der Name einer Laufvariablen aus einem einzelnen Buchstaben bestehen).

Textvariable

Mit dem folgenden Programm können Sie einen Freund zu einem kleinen Ratespiel auffordern, das auch unter dem Namen „Galgenmännchen" bekannt ist. Beim Spiel „Galgenmännchen" gilt es, ein Rätselwort allmählich zu entschlüsseln, indem nach dem Vorhandensein bestimmter Buchstaben gefragt wird. Der Homecomputer teilt jeweils ein Lösungswort mit, das die bereits entdeckten Buchstaben beinhaltet.

Zum Beispiel könnte man als Rätselwort „HIMBEER-EIS" nehmen.

Löschen Sie zunächst mit

→ NEW das alte Programm und geben Sie das Programm „Galgenmännchen" aus Tab. 6 ein. In dieser Tabelle ist für jeden Rechner das vollständige Listing abgedruckt.

Starten Sie es mit

→ RUN und geben sie dann
→ Buchstaben Ihrer Wahl ein.

Im Programm taucht eine neue Variablenart auf. Sie nimmt Text auf im Gegensatz zu der uns bisher bekannten Zahlvariablen und kann deshalb Textvariable genannt werden.

Der Name einer Textvariablen hört immer mit dem Zeichen $ auf (Tab. 7).

Im „Galgenmännchen"-Programm kommen drei Textvariable vor: R$, L$ und B$. Die Namen sind entsprechend der Bedeutung den Variablen als Rätselwort, Lösungswort und Buchstabe gewählt (jeweils der Anfangsbuchstabe). Auf diese Weise sich selbst erläuternde Variablennamen zu wählen, erleichtert es sehr, fremde Programme zu verstehen oder sich in eigene Programme nach längerer Zeit wieder hineinzufinden.

Der BASIC-Kurs wird im nächsten Heft fortgesetzt.

Spectrum	VC-20	TI-99/4A	Atari 600XL
			5 DIM R$(10) 6 DIM L$(16) 7 DIM B$(1)
10 LET R$ = „HIMBEER-EIS" 20 LET L$ = "-----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF R$(I) = B$ THEN LET L$(I) = B$	10 LET R$ = „HIMBEER-EIS" 20 LET L$ = "-----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF MID$(R$,I,1)=B$ THEN LET L$=LEFT$(L$,I-1)+B$+RIGHT$(I$,16-I)	10 LET R$ = „HIMBEER-EIS" 20 LET L$ = "-----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF SEG$(R$,I,1)=B$ THEN 55 ELSE 60 55 LET L$=SEG$(L$,1,I-1)&B$&SEG$(L$,I+1,16-I)	10 LET R$ = „HIMBEER-EIS" 20 LET L$ = "-----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF R$(I,I)=B$ THEN LET L$(I,I)=B$
60 NEXT I 70 PRINT L$ (TO LEN(R$))	60 NEXT I 70 PRINT LEFT$(L$,LEN(R$))	60 NEXT I 70 PRINT SEG$(L$,1,LEN(R$))	60 NEXT I 70 PRINT L$(1,LEN(R$))
80 GOTO 30	80 GOTO 30	80 GOTO 30	80 GOTO 30

Tab. 6. Programm „Galgenmännchen"

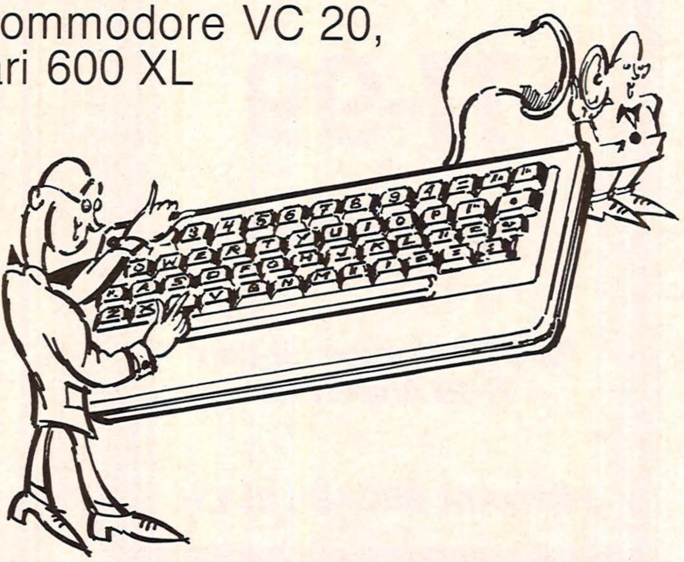
Spectrum	VC-20	TI-99/4A	Atari 600XL
einzelner Buchstabe gefolgt vom Zeichen $	wie eine Zahlvariable, jedoch maximal drei Zeichen; letztes Zeichen ein $	wie eine Zahlvariable; letztes Zeichen ein $	unbegrenzt; letztes Zeichen ein $

Tab. 7. zugelassene Namen von Textvariablen

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC 20, Texas Instruments 99/4A und Atari 600 XL



Am Ende der vorigen Folge dieses Kurses stand das Programm „Galgenmännchen" (Tab. 6). In ihm tauchten zum ersten Mal Textvariable auf. Die Listings für die vier Rechner zeigen, daß die Verarbeitung von Text jeweils etwas anders gehandhabt wird.

Gemeinsam ist allen vier Listings, daß zunächst durch INPUT B$ ein Buchstabe eingegeben wird und danach eine FOR-NEXT-Schleife mit der Laufvariablen I abgearbeitet wird. Die Laufvariable I hat dabei die Aufgabe, nacheinander auf alle Buchstaben des Rätselwortes zu zeigen. Die obere Grenze für die Laufvariable ist LEN(R$). Dies ist die Länge des Rätselwortes. Das Programm setzt beim Rätselwort „HIMBEEREIS" dort also die Zahl 10 ein. Man kann sich somit ersparen, bei jedem neuen Rätselwort die obere Grenze der Laufvariable I ändern zu müssen.

In Zeile 50 wird jeweils ein LET-Befehl nur unter einer bestimmten Bedingung ausgeführt. Diese Bedingung lautet: „Wenn der I-te Buchstabe des Rätselwortes der abgefragte Buchstabe B$ ist".

Unterschiedliche Textbehandlung

Daß der I-te Buchstabe des Rätselwortes gemeint ist, muß man jedem Rechner auf andere Weise klarmachen:

Spectrum	R$(I)
VC20	MID$(R$,I,1)
TI-99/4A	SEG$(R$,I,1)
Atari 600XL	R$(I,I)

Beim Herausgreifen eines Teiles aus einem Text muß also bei manchen Rechnern die Stelle des Anfangsbuchstaben und des Endbuchstaben, bei anderen Anfangsstelle und Länge des Teiles angegeben werden. Beim Spectrum reicht es bei einem Einzelbuchstaben, einmal die Stelle anzugeben. Bei größeren Teilstücken müssen auch beim Spectrum Anfangs- und Endstelle mit TO dazwischen mitgeteilt werden; wenn eine von beiden weggelassen wird, setzt der Spectrum automatisch 1 bzw. die ganze Länge des Textes ein.

Der VC20 kennt außer MID$ noch die Funktionen LEFT$ und RIGHT$, mit LEFT$ kann der linke Teil eines Textes in bestimmter Länge herausgegriffen werden, mit RIGHT$ kann das gleiche für den rechten Teil eines Textes getan werden. Weitere Unterschiede zwischen den Rechnern werden direkt aus den Listings heraus verständlich.

In allen vier Listings wird in Programmzeile 50 (und 55 beim TI-99/4A) der I-ten Stelle des Lösungswortes der abgefragte Buchstabe B$ zugewiesen, wenn er beim Rätselwort an dieser Stelle auftaucht. In Zeile 70 wird das mehr oder weniger vollständige Lösungswort gedruckt, und in Zeile 80 wird zur Eingabe des nächsten Buchstaben zurückgesprungen.

Dialogmöglichkeiten

Mit dem INPUT-Befehl haben wir eine Möglichkeit kennengelernt, in einem Dialog mit dem Home-Computer einen Programmablauf zu steuern. Zu einem echten Dialog fehlt eigentlich nur, daß der Home-Computer vor der Eingabe des Benutzers eine entsprechende Frage stellt. Im Galgenmännchen-Programm könnte diese Frage in der Programmzeile → 30 PRINT „WELCHER

Scan: S. Höltgen

BUCHSTABE?" stehen (Sie müssen nach Eingabe dieser Programmzeile auch die Zeile → 35 INPUT B$ eingeben, weil 30 INPUT B$ überschrieben wurde.

Wir wollen nun eine weitere Dialogmöglichkeit besprechen. Sie besteht darin, daß das Programm die Tastatur danach abfragt, ob eine und welche Taste gerade gedrückt wird (Tab. 8). Im Unterschied zum INPUT-Befehl hält das Programm dabei nicht an, es ist auch kein Tastendruck zur Markierung des Endes der Eingabe nötig. Die Tastatur wird nur einmal kurz abgefragt, und danach wird im weiteren Programm fortgefahren.

Wenn man sowieso nur ein einzelnes Zeichen eingeben will, spart man also im Vergleich zu INPUT mit der Tastaturabfrage einen Tastendruck. Ein Problem scheint zunächst zu sein, als Dialogpartner den Zeitpunkt der Tastaturabfrage zu erwischen. Wie man dieses Problem einfach löst, wird sich in dem folgenden „Schatzsuche"-Programm zeigen (Tab. 9). Dort wird in den Programmzeilen 70 und 80 eine Tastaturabfrage ständig wiederholt, solange noch keine Taste gedrückt wurde. Starten Sie nach dem Eintippen das Programm mit
→ RUN und drücken Sie danach
→ eine der Tasten I, J, K oder M, um den Schatz zu suchen.

Wir wollen die Schatzsuche jetzt Zeile für Zeile analysieren. Zunächst einmal wird der Schatz versteckt. Dafür steht eine 14 Spalten breite und 21 Zeilen hohe Fläche am linken oberen Bildschirmrand zur Verfügung. Der Ort wird durch Berechnung einer Schatzzeile SZ und einer Schatzspalte SS in den Programmzeilen 10 bis 22 festgelegt.

Zufallszahlen

Damit die Schatzsuche überhaupt interessant ist, soll natürlich niemand (auch der Schreiber des Programmes nicht) wissen, wo sich der Schatz befindet. Dazu wählt der Home-Computer eine Zahl zwischen 0 (einschließlich der 0) und 1 (ohne die 1) zufällig aus. RND (RND[1] beim VC20) steht stellvertretend für diese Zahl. Prüfen Sie diese Behauptung nach, indem Sie mehrmals hintereinander das Kommando
→ PRINT RND
...(PRINT RND [1] beim VC20) eingeben (ganz nebenbei können Sie sehen, daß Ihr Heimcomputer Dezimalzahlen mit einem Dezimalpunkt und nicht mit einem Dezimalkomma schreibt).

Auch für die Eingabe einer Dezimalzahl verlangt er diese Schreibweise von Ihnen. Durch Multiplikation mit 21 und Addition von 1 entsteht aus RND (RND[1] beim VC20) eine zufällige Zahl zwischen 1 (einschließlich der 1) und 21 (ohne die 21). Im allgemeinen wird es sich dabei um keine ganze Zahl handeln. Als Schatzzeilennummer können wir jedoch nur eine ganze Zahl gebrauchen. INT macht aus einer Dezimalzahl die nächstkleinere ganze Zahl (falls schon eine ganze Zahl vorlag, bleibt es nach INT bei dieser ganzen Zahl).

In den Programmzeilen 10 und 12 wird also der Variablen SZ eine ganze Zahl zwischen 1 (einschließlich der 1) und 21 (einschließlich der 21) zugewiesen. In den Programmzeilen 20 und 22 erhält die Variable SS eine ganze Zahl zwischen 1 (einschließlich der 1) und 14 (einschließlich der 14). In den Programmzeilen 30 und 40 wird festgelegt, daß der Sucher seinen Weg in der Bildschirmzeile Z = 11 und der Spalte S = 7 beginnt, also ungefähr in der Mitte der Fläche.

Schatz gefunden?

In Programmzeile 50 wird die momentane Position des Suchers auf dem Bildschirm sichtbar gemacht. In Programmzeile 60 wird überprüft, ob der Sucher den Schatz schon gefunden hat (möglicherweise sitzt er ja schon ganz am Anfang auf dem Schatz), woraufhin die Suche beendet ist. Dies übernimmt wieder eine bedingte Anweisung. Die Anweisung ist ein Vorwärtssprung nach Programmzeile 180. Die Bedingung besteht aus zwei Teilen. Es müssen sowohl die Zeile des Suchers und die Schatzzeile als auch die Spalte des Suchers und die Schatzspalte übereinstimmen. Die Verknüpfung beider Bedingungen durch „sowohl-als auch" übernimmt im Programm AND.

Da geht's lang

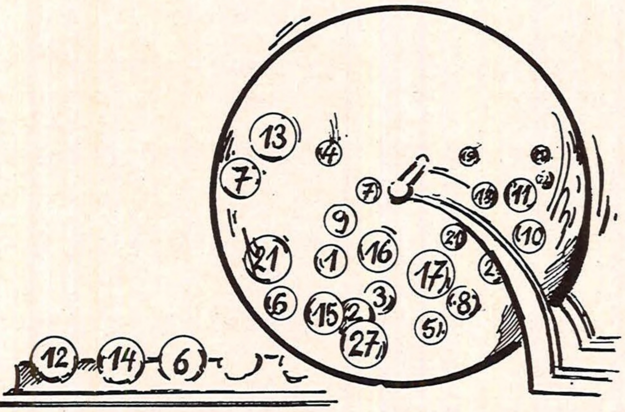
In den Programmzeilen 70 und 80 wird (wie schon erwähnt) eine Tastaturabfrage so lange wiederholt, bis endlich ein Tastendruck vorliegt. In Programmzeile 85 markiert der Sucher hinter sich die schon betretenen Stellen.

Die Textvariable R$ legt die Richtung fest, in die der Sucher sich bewegt. Für Bewegungen auf dem Bildschirm eignen sich die Tasten I, J, K und M wegen ihrer benachbarten Lage gut. In den Programmzeilen 90 bis 120 werden entsprechend der Positionen dieser vier Tasten innerhalb der Tastatur die Zeilennummer Z oder die Spaltennummer S um 1 vergrößert oder verringert. Dies führt dann zu einer Bewegung nach Norden, Westen, Osten oder Süden.

In den Programmzeilen 130 bis 165 wird verhindert, daß der Sucher die Fläche verläßt. Falls die Zeilennummer oder die Spaltennummer einen Wert annehmen, der außerhalb der erlaubten Fläche liegt, macht der Sucher sofort eine Kehrtwendung, bevor sein „Fehltritt" auf dem Bildschirm sichtbar werden könnte.

In Programmzeile 170 wird durch einen Rückwärtssprung ein neuer Suchschritt ausgeführt. Die Programmzeile 180 schließlich zeigt durch einen Stern den Fundort an.

Der BASIC-Kurs wird im nächsten Heft fortgesetzt.



Spectrum	VC 20	TI-99/4A	Atari 600 XL
INKEY$	GET A$	CALL KEY(0,K,ST)	OPEN#1,4,0,"K:" GET#1,K

Tab. 8. Tastaturabfrage

Scan: S. Höltgen

Spectrum	VC 20	TI-99/4A	Atari 600 XL
10 LET R$ = "HIMBEER EIS" 20 LET L$ = "----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF R$(I) = B$ THEN LET L$(I) = B$ 60 NEXT I 70 PRINT L$ (TO LEN(R$)) 80 GOTO 30	10 LET R$ = "HIMBEER EIS" 20 LET L$ = "----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF MID$(R$,I,1) = B$ THEN LET L$ = LEFT$(L$,1−1) +B$+RIGHT$(L$,16−I) 60 NEXT I 70 PRINT LEFT$ (L$,LEN(R$)) 80 GOTO 30	10 LET R$ = "HIMBEER EIS" 20 LET L$ = "----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF SEG$(R$,I,1) = B$ THEN 55 ELSE 60 55 LET L$ = SEG$(L$,1,I−1) &B$&SEG$(L$,I+1,16−I) 60 NEXT I 70 PRINT SEG$(L$,1,LEN (R$)) 80 GOTO 30	5 DIM R$(10) 6 DIM L$(16) 7 DIM B$(1) 10 LET R$ = "HIMBEER EIS" 20 LET L$ = "----------------" 30 INPUT B$ 40 FOR I = 1 TO LEN(R$) 50 IF R$(I,I) = B$ THEN LET L$(I,I) = B$ 60 NEXT I 70 PRINT L$(1,LEN(R$)) 80 GOTO 30

Tab. 6. Programm „Galgenmännchen"

Spectrum	VC 20	TI-99/4A	Atari 600 XL
10 LET SZ=RND*21+1 12 LET SZ=INT(SZ) 20 LET SS=RND*14+1 22 LET SS=INT(SS) 30 LET Z=11 40 LET S=7 50 PRINT AT Z, S;"+" 60 IF Z=SZ AND S=SS THEN GOTO 180 70 LET R$=INKEY$ 80 IF R$="" THEN GOTO 70 85 PRINT AT Z,S;"−" 90 IF R$="M" THEN LET Z=Z+1 100 IF R$="I" THEN LET Z=Z−1 110 IF R$="K" THEN LET S=S+1 120 IF R$="J" THEN LET S=S−1 130 IF Z<1 THEN LET Z=Z+1 140 IF Z>21 THEN LET Z=Z−1 150 IF S<1 THEN LET S=S+1 160 IF S>14 THEN LET S=S−1 170 GOTO 50 180 PRINT AT Z,S;"*"	10 LET SZ=RND(1)*21+1 12 LET SZ=INT(SZ) 20 LET SS=RND(1)*14+1 22 LET SS=INT(SS) 30 LET Z=11 40 LET S=7 50 SPC(Z):TAB(S):PRINT"+" 60 IF Z=SZ AND S=SS THEN GOTO 180 70 GET R$ 80 IF R$="" THEN GOTO 70 85 SPC(Z):TAB(S):PRINT"−" 90 IF R$="M" THEN LET Z=Z+1 100 IF R$="I" THEN LET Z=Z−1 110 IF R$="K" THEN LET S=S+1 120 IF R$="J" THEN LET S=S−1 130 IF Z<1 THEN LET Z=Z+1 140 IF Z>21 THEN LET Z=Z−1 150 IF S<1 THEN LET S=S+1 160 IF S>14 THEN LET S=S−1 170 GOTO 50 180 SPC(Z):TAB(S):PRINT"*"	10 LET SZ= RND*21+1 12 LET SZ=INT(SZ) 20 LET SS=RND*14+1 22 LET SS=INT(SS) 30 LET Z=11 40 LET S=7 50 CALL HCHAR(Z,S,43,1) 60 IF Z=SZ THEN 65 ELSE 70 65 IF S=SS THEN 180 ELSE 70 70 CALL KEY (0,K,ST) 80 IF ST=0 THEN 70 85 CALL HCHAR(Z,S,45) 90 IF K=109 THEN 95 ELSE 100 95 LET Z=Z+1 100 IF K=105 THEN 105 ELSE 110 105 LET Z=Z−1 110 IF K=107 THEN 115 ELSE 120 115 LET S=S+1 120 IF K=106 THEN 125 ELSE 130 125 LET S=S−1 130 IF Z<1 THEN 135 ELSE 140 135 LET Z=Z+1 140 IF Z>21 THEN 145 ELSE 150 145 LET Z=Z−1 150 IF S<1 THEN 155 ELSE 160 155 LET S=S+1 160 IF S>14 THEN 165 ELSE 170 165 LET S=S−1 170 GOTO 50 180 CALL HCHAR(Z,S,42,1)	5 OPEN#1,4,0,"K:" 10 LET SZ=RND(1)*21+1 12 LET SZ=INT(SZ) 20 LET SS=RND(1)*14+1 22 LET SS=INT(SS) 30 LET Z=11 40 LET S=7 50 POS.S,Z:PRINT"+" 60 IF Z=SZ AND S=SS THEN GOTO 180 70 GET #1,K 80 IF K=0 THEN GOTO 70 85 POS.S,Z:PRINT"−" 90 IF K=77 THEN LET Z=Z+1 100 IF K=73 THEN LET Z=Z−1 110 IF K=75 THEN LET S=S+1 120 IF K=74 THEN LET S=S−1 130 IF Z<1 THEN LET Z=Z+1 140 IF Z>21 THEN LET Z=Z−1 150 IF S<1 THEN LET S=S+1 160 IF S>14 THEN LET S=S−1 170 GOTO 50 180 POS.S,Z:PRINT"*"

Tab. 9. Programm „Schatzsuche"

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC 20, Texas Instruments 99/4A und Atari 600 XL

Primzahlen

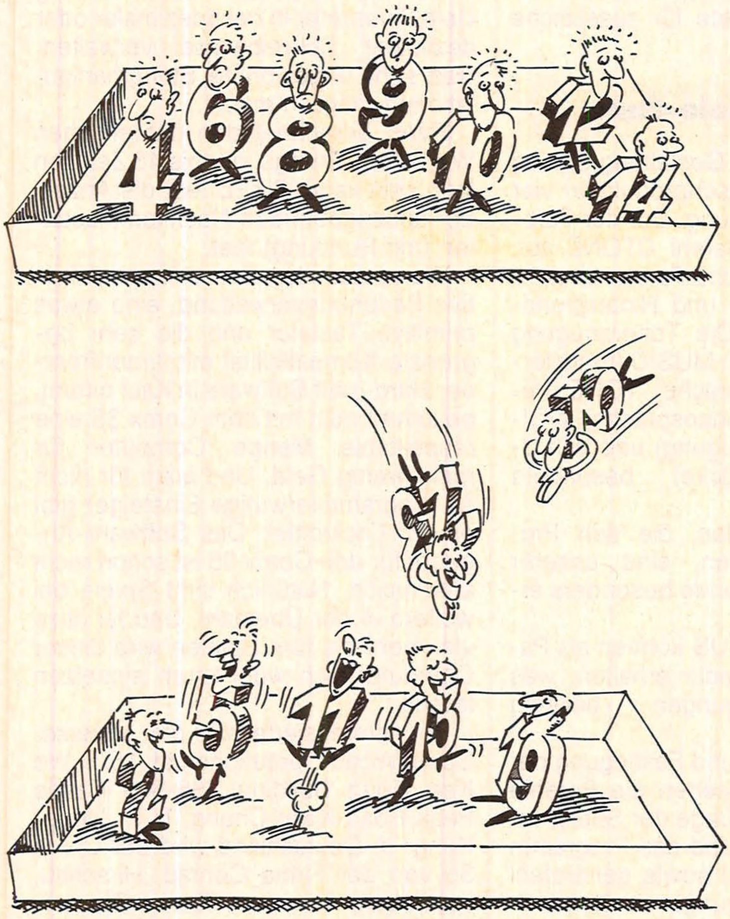
Wie kann ein Programm überprüfen, ob 95 eine Primzahl ist oder nicht?

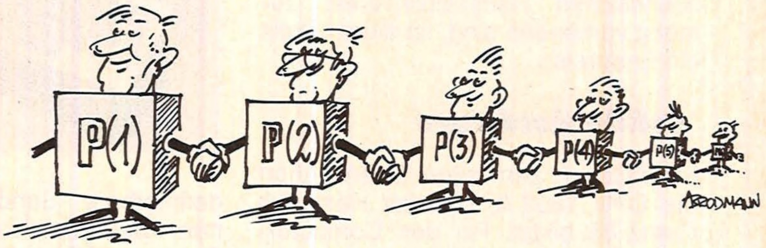
Eine Primzahl läßt sich bekanntlich durch keine ganze Zahl ohne Rest teilen, nur durch sich selbst und 1. Es genügt, die Teilbarkeit von 95 durch 2, 3, 5 und 7 zu überprüfen.

Wie ein Programm Teilbarkeit feststellen kann, wollen wir am Beispiel der Primzahl 2 untersuchen. Zunächst wird 95 durch 2 geteilt:
95/2 = 47.5

Teilbarkeit durch 2 würde vorliegen, wenn dabei eine ganze Zahl (zum Beispiel 48) herausgekommen wäre.

Um in einem Programm feststellen zu können, ob eine Zahl





ganz ist, hilft uns die Funktion INT. Wenn man nämlich INT auf eine ganze Zahl anwendet, ist das Ergebnis wieder die gleiche ganze Zahl. Nur wenn man INT auf eine nicht-ganze Zahl anwendet, weicht das Ergebnis von dieser Zahl ab.

2 ist also kein Teiler von 95, weil 97/2 <> INT (97/2)
(= 48.5) (= 48)

5 ist Teiler von 95, denn
95/5 = INT (95/5)
(= 19) (= 19)

95 ist also keine Primzahl, denn wir haben einen Teiler von 95 gefunden.

Wie wir die nötigen Rechenschritte in einem Programm durchführen können, zeigt die Tabelle 10. Die zu testenden Zahlen (bis 168) können nach RUN eingegeben werden.

Für die ersten fünf Primzahlen werden im Programm die Variablennamen P1, P2, P3, P4 und P5 benutzt. Auffällig ist, daß für alle fünf Primzahlen völlig gleichartige Anweisungen ausgeführt werden sollen, nämlich die bedingten Anweisungen in den Zeilen 70 bis 110. Um uns diese Übereinstimmung zunutze machen zu können, steht uns ein anderer Variablentyp zur Verfügung: das Variablenfeld.

Felder

Alle fünf Primzahlen können in einem Feld mit dem Namen P zusammengefaßt werden. Mit der Anweisung

```
5 DIM P(5)
```

wird dieses Feld vereinbart. Die einzelnen Primzahlen sind jetzt P(1), P(2), P(3), P(4) und P(5). Dadurch werden die einzelnen Primzahlen für eine recht kurze Schleife ansprechbar. Diese Schleife hat die Form:

```
70 FOR I=1 TO 5
80 IF Z/P(I)=INT(Z/P(I))
   THEN GOTO 140
90 NEXT I
```

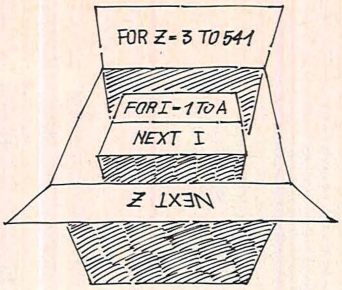
Damit haben wir die Zeilen 70 bis 90 überschrieben. Die Zeilen 100 und 110 sind jetzt überflüssig und können gelöscht werden. Einzelne Zeilen löscht man, indem man nur deren Zeilennummern eingibt:

```
100
110
```

Jetzt müssen nur noch die Zeilen 10 bis 50 auf die Verwendung des Variablenfeldes umgestellt werden:

```
10 LET P(1)=2
20 LET P(2)=3
30 LET P(3)=5
40 LET P(4)=7
```

Scan: S. Höltgen



50 LET P(5)=11

Überprüfen Sie mit RUN, daß das neue Programm die gleichen Ergebnisse liefert wie das vorherige.

Verschachtelte Schleifen

Unsere momentane Primzahlprogrammversion kann nur für die Zahlen 2 bis 168 eine Entscheidung treffen. Für größere Zahlen müßte das Programm vorher schon mehr als nur die ersten fünf Primzahlen kennen. Je größer die Testzahl ist, um so mehr Primzahlen muß das Programm vorher auch schon parat haben. Es ist also ein Programm erforderlich, das sich ein Feld von Primzahlen aufbaut und dieses selbständig nach oben hin weiter ergänzt. Als Starthilfe wollen wir dem Programm nur die allererste Primzahl P(1)=2 zur Verfügung stellen. In Tabelle 11 ist das vollständige neue Programm abgedruckt. Versuchen Sie als kleine Übung, es durch Ergänzen und Verändern des vorigen Programmes aufzubauen. Starten Sie es danach mit RUN.

Wenden wir uns nun den Einzelheiten des Programmes zu. A ist die Anzahl der Zahlen, die bereits als Primzahlen entlarvt wurden. Ein wichtiger neuer Aspekt dieses Programmes ist es, daß hier zwei FOR-NEXT-Schleifen ineinander verschachtelt sind.

Die innere Schleife mit der Laufvariablen I überprüft für die augenblickliche Testzahl Z, ob eine der bisherigen Primzahlen Teiler von Z ist. Diese Überprüfung muß erst vollkommen abgeschlossen sein, bevor die nächste Testzahl untersucht wird. Mit anderen Worten: Die innere Schleife wird erst vollkommen abgearbeitet, bevor die Laufvariable Z der äußeren Schleife verändert wird. Damit die Laufvariablen der inneren Schleife und der äußeren Schleife sich unabhängig voneinander ändern können, müssen sie natürlich verschiedene Namen haben.

Wenn der Test ergibt, daß die Testzahl eine Primzahl ist, wird in den Zeilen 110 bis 130 die Anzahl der Primzahlen um eins erhöht, die neue Primzahl als die A-te ins Feld P aufgenommen und als P(A) ausgedruckt.

Spectrum und VC 20	TI-99/4A	Atari 600 XL
10 LET P1=2	10 LET P1=2	10 LET P1=2
20 LET P2=3	20 LET P2=3	20 LET P2=3
30 LET P3=5	30 LET P3=5	30 LET P3=5
40 LET P4=7	40 LET P4=7	40 LET P4=7
50 LET P5=11	50 LET P5=11	50 LET P5=11
60 INPUT Z	60 INPUT Z	60 INPUT Z
65 IF Z=1 THEN GOTO 140	65 IF Z=1 THEN GOTO 140	65 IF Z=1 THEN GOTO 140
70 IF Z/P1=INT(Z/P1) AND Z<>P1 THEN GOTO 140	70 IF Z/P1<>INT(Z/P1) THEN 80 75 IF Z<>P1 THEN 140	70 IF Z/P1=INT(Z/P1) AND Z<>P1 THEN GOTO 140
80 IF Z/P2=INT(Z/P2) AND Z<>P2 THEN GOTO 140	80 IF Z/P2<>INT(Z/P2) THEN 90 85 IF Z<>P2 THEN 140	80 IF Z/P2=INT(Z/P2) AND Z<>P2 THEN GOTO 140
90 IF Z/P3=INT(Z/P3) AND Z<>P3 THEN GOTO 140	90 IF Z/P3<>INT(Z/P3) THEN 100 95 IF Z<>P3 THEN 140	90 IF Z/P3=INT(Z/P3) AND Z<>P3 THEN GOTO 140
100 IF Z/P4=INT(Z/P4) AND Z<>P4 THEN GOTO 140	100 IF Z/P4<>INT(Z/P4) THEN 110 105 IF Z<>P4 THEN 140	100 IF Z/P4=INT(Z/P4) AND Z<>P4 THEN GOTO 140
110 IF Z/P5=INT(Z/P5) AND Z<>P5 THEN GOTO 140	110 IF Z/P5<>INT(Z/P5) THEN 120 115 IF Z<>P5 THEN 140	110 IF Z/P5=INT(Z/P5) AND Z<>P5 THEN GOTO 140
120 PRINT"PRIMZAHL"	120 PRINT"PRIMZAHL"	120 PRINT"PRIMZAHL"
130 GOTO 60	130 GOTO 60	130 GOTO 60
140 PRINT"KEINE PRIMZAHL"	140 PRINT"KEINE PRIMZAHL"	140 PRINT"KEINE PRIMZAHL"
150 GOTO 60	150 GOTO 60	150 GOTO 60

Tab. 10. Primzahl-Programm ohne Feld

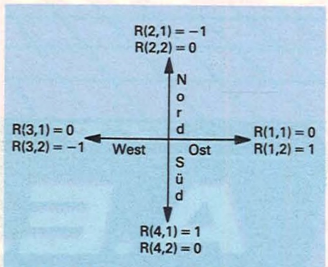
Spectrum und VC 20	TI-99/4A	Atari 600 XL
5 DIM P(100)	5 DIM P(100)	5 DIM P(100)
10 LET P(1)=2	10 LET P(1)=2	10 LET P(1)=2
20 LET A=1	20 LET A=1	20 LET A=1
30 PRINT 2	30 PRINT 2	30 PRINT 2
60 FOR Z=3 TO 541	60 FOR Z=3 TO 541	60 FOR Z=3 TO 541
70 FOR I=1 TO A	70 FOR I=1 TO A	70 FOR I=1 TO A
80 IF Z/P(I)=INT(Z/P(I)) THEN GOTO 140	80 IF Z/P(I)=INT(Z/P(I)) THEN 140	80 IF Z/P(I)=INT(Z/P(I)) THEN GOTO 140
90 NEXT I	90 NEXT I	90 NEXT I
110 LET A=A+1	110 LET A=A+1	110 LET A=A+1
120 LET P(A)=Z	120 LET P(A)=Z	120 LET P(A)=Z
130 PRINT P(A)	130 PRINT P(A)	130 PRINT P(A)
140 NEXT Z	140 NEXT Z	140 NEXT Z

Tab. 11. Primzahl-Programm mit Feld

Der BASIC-Kurs wird im nächsten Heft fortgesetzt

Scan: S. Höltgen

Programmieren in BASIC



Feld R für vier Richtungen

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC 20, Texas Instruments 99/4A und Atari 600XL

Zweidimensionale Variablenfelder

Ein Beispiel für ein zweidimensionales Variablenfeld ist die Suchfläche des Programmes „Schatzsuche" aus der vorherigen Folge des Kurses. Um ein Element dieser Fläche herauszugreifen, müssen eine Zeilenzahl und eine Spaltenzahl angegeben werden (man bezeichnet diese beiden auch als die Koordinaten oder die Indizes des Feldes). Im Schatzsucheprogramm war die Vereinbarung eines Feldes jedoch überflüssig. Das Programm benötigte nämlich nirgends die Information, ob an dem momentanen Standort schon einmal gesucht worden war oder nicht.

Erforderlich ist die Benutzung eines Feldes beim Spiel „Solitaire". Das Spielfeld dieses Spiels besteht aus einem 9×9-Quadrat, dem aber in jeder Ecke ein 3×3-Quadrat fehlt. Alle Einzelfelder sind mit Steinen besetzt. Nur das Mittelfeld ist frei. Ein Stein darf einen benachbarten Stein (ein Feld hat vier Nachbarn) überspringen, wenn das Feld dahinter frei ist. Der übersprungene Stein wird danach entfernt. Ziel eines Spielers ist es, daß am Ende möglichst wenige Steine übrig bleiben.

Um später im Programm besser beschreiben zu können, daß ein Stein sich nicht mehr auf dem Spielfeld befindet, vereinbart man im Programm ein etwas größeres Feld (11×11-Feld) mit der Anweisung DIM F$(11,11). Die 1. Koordinate des Feldes soll gleich der Zeilenzahl und die 2. Koordinate gleich der Spaltenzahl der Bildschirmdarstellung sein. Mit F$(6,8) ist also zum Beispiel das Einzelfeld in der 6. Zeile und der 8. Spalte gemeint.

F$ ist ein Textvariablenfeld. Als „Spielfeld" eignet es sich besser als ein Zahlvariablenfeld, weil ein Zeichen genügt, um den „Zustand" eines Einzelfeldes zu beschreiben. Wir legen fest: "X" soll ein mit einem Stein besetztes Einzelfeld bezeichnen, "0" ein verbotenes Feld außerhalb des Spielfeldes und " " ein Leerfeld innerhalb des Spielfeldes.

Das Spielfeld von „Solitaire"

Wir wollen ein Programm entwickeln, das den Computer in die Lage versetzt, selbständig Solitaire zu spielen. Der erste Teil soll das Spielfeld in seiner anfänglichen Besetzung auf dem Bildschirm abbilden. Er reicht bis Programmzeile 140 des Listings aus Tabelle 12. Tippen Sie diesen Teil zunächst ohne die Zeilen 50 bis 95 und 130 bis 140 ein und starten Sie ihn mit RUN.

Das eigentliche Spielfeld hat aber viel weniger Steine. Wir wollen zunächst dafür sorgen, daß die linke obere Ecke nicht zum Spielfeld gehört. Die Elemente, die sich dort befinden, sollen das Zeichen "0" zugewiesen bekommen. Ihnen allen ist gemeinsam, daß die Zeilenzahl und die Spaltenzahl kleiner als 5 sind. Weiterhin soll die Laufvariable I die Zeilenzahl und die Laufvariable J die Spaltenzahl angeben. Folgende Programmzeile bewirkt dann, daß ein in der linken oberen Ecke zunächst als besetzt gemeldetes Element nachträglich als außerhalb des Spielfeldes vermerkt wird:

```
50 IF I<5 AND J<5 THEN LET
F$(I,J) = "0"
```

Ein RUN nach Eingabe dieser Zeile zeigt sofort, wie sie sich auswirkt.

Die Bedingungen für die anderen drei Ecken lassen sich ebenfalls leicht finden. An jedem Flügel des Spielfeldes ist jetzt noch der äußere Rand zuviel. Für ihn gilt, daß entweder die Zeilenzahl oder die Spaltenzahl den Wert 1 oder 11 annehmen. Durch entsprechende Bedingungen ab Zeile 90 wird dieser Rand abgeschnitten.

Schließlich wird in den Zeilen 130 und 140 das mittlere Feld freigemacht. Im weiteren Verlauf (ab Zeile 200) sucht das Programm nach Zügen und führt einen aus.
In den Programmzeilen 310 bis 360 wird ein Element des inneren 9×9-Feldes zufällig gewählt und eventuell versucht, in eine ebenfalls zufällig ausgewählte Richtung zu springen.

Es kann nun drei Gründe geben, weshalb ein Zug nicht ausgeführt werden kann; es wird in diesen Fällen der Zug neu bestimmt. Die drei möglichen Gründe sind:
– es befindet sich an der Stelle gar kein Stein (Zeile 330)
– es wird in der Richtung kein Stein übersprungen (Zeile 350)
– hinter dem übersprungenen Stein ist kein Feld leer (Zeile 360)

In alle Himmelsrichtungen

Die vier Richtungen werden durch die Programmzeilen 200 bis 280 im Feld R gespeichert; es handelt sich um ein zweidimensionales Zahlvariablenfeld. Sein Aufbau ist nicht so leicht einsichtig, da es nicht wie F$ als Spielfeld unmittelbar verstanden werden kann. Seine 1. Koordinate numeriert die vier Richtungen durch (Reihenfolge: Ost-Nord-West-Süd); seine 2. Koordinate gibt an, welche Koordinate des Feldes F$ gemeint ist (1 entspricht Zeile, 2 entspricht Spalte).

Der Wert eines einzelnen Elements von R legt fest, wie sich die angesprochene Koordinate von F$ ändert, wenn man sich im Spielfeld um ein Einzelfeld in der betreffenden Richtung bewegt.

R(3,2)=−1 bedeutet also: Die Spaltenzahl verringert sich um eins, wenn man sich im Spielfeld um ein Einzelfeld nach Westen bewegt. Die Zuordnung aller Elemente von R zu den vier Richtungen zeigt die Zeichnung.

In Zeile 360 muß bei der Abfrage, ob das Einzelfeld hinter dem übersprungenen Stein frei ist, zu jeder Koordinate von F das Doppelte des jeweiligen

Spectrum	VC20	TI-99/4A	Atari 600 XL
	5 LET C$= "SQQQQQQQQQQ"		
10 DIM F$(11,11)	10 DIM F$(11,11)	10 DIM F$(11,11)	10 DIM F$(121)
20 FOR I=1 TO 11	20 FOR I=1 TO 11	20 FOR I=1 TO 11	20 FOR I=1 TO 11
30 FOR J=1 TO 11	30 FOR J=1 TO 11	30 FOR J=1 TO 11	30 FOR J=1 TO 11
40 LET F$(I,J)="X"	40 LET F$(I,J)="X"	40 LET F$(I,J)="X"	40 LET F$(11*I+J-11)="X"
50 IF I<5 AND J<5 THEN LET F$(I,J)="0" 60 IF I<5 AND J>7 THEN LET F$(I,J)="0" 70 IF I>7 AND J<5 THEN LET F$(I,J)="0" 80 IF I>7 AND J>7 THEN LET F$(I,J)="0" 90 IF I=1 OR I=11 OR J=1 OR J=11 THEN LET F$(I,J)="0"	50 IF I<5 AND J<5 THEN LET F$(I,J)="0" 60 IF I<5 AND J>7 THEN LET F$(I,J)="0" 70 IF I>7 AND J<5 THEN LET F$(I,J)="0" 80 IF I>7 AND J>7 THEN LET F$(I,J)="0" 90 IF I=1 OR I=11 OR J=1 OR J=11 THEN LET F$(I,J)="0"	50 IF I>=5 THEN 80 60 IF J<5 THEN 95 70 IF J>7 THEN 95 80 IF I<=7 THEN 90 82 IF J<5 THEN 95 84 IF J>7 THEN 95 90 IF I=1 THEN 95 91 IF I=11 THEN 95 92 IF J=1 THEN 95 93 IF J<>11 THEN 100 95 LET F$(I,J)="0"	50 IF I<5 AND J<5 THEN LET F$(11*I+J-11)="0" 60 IF I<5 AND J>7 THEN LET F$(11*I+J-11)="0" 70 IF I>7 AND J<5 THEN LET F$(11*I+J-11)="0" 80 IF I>7 AND J>7 THEN LET F$(11*I+J-11)="0" 90 IF I=1 OR I=11 OR J=1 OR J=11 THEN LET F$(11*I+J-11)="0"
100 PRINT AT I,J; F$(I,J)	100 PRINT LEFT$(C$,I); TAB(J);F$(I,J)	100 LET X=ASC(F$(I,J)) 101 CALL HCHAR (I,J,X)	100 POS.J,I:PRINT F$(11*I +J-11)
110 NEXT J	110 NEXT J	110 NEXT J	110 NEXT J
120 NEXT I	120 NEXT I	120 NEXT I	120 NEXT I
130 LET F$(6,6)=" "	130 LET F$(6,6)=" "	130 LET F$(6,6)=" "	130 LET F$(61,61)=" "
140 PRINT AT 6,6;" "	140 PRINT LEFT$(C$,6); TAB(6);" "	140 CALL HCHAR(6,6,32)	140 POS.6,6:PRINT" "
200 DIM R(4,2)	200 DIM R(4,2)	200 DIM R(4,2)	200 DIM R(4,2)
210 LET R(1,1)=0	210 LET R(1,1)=0	210 LET R(1,1)=0	210 LET R(1,1)=0
220 LET R(1,2)=1	220 LET R(1,2)=1	220 LET R(1,2)=1	220 LET R(1,2)=1
230 LET R(2,1)=-1	230 LET R(2,1)=-1	230 LET R(2,1)=-1	230 LET R(2,1)=-1
240 LET R(2,2)=0	240 LET R(2,2)=0	240 LET R(2,2)=0	240 LET R(2,2)=0
250 LET R(3,1)=0	250 LET R(3,1)=0	250 LET R(3,1)=0	250 LET R(3,1)=0
260 LET R(3,2)=-1	260 LET R(3,2)=-1	260 LET R(3,2)=-1	260 LET R(3,2)=-1
270 LET R(4,1)=1	270 LET R(4,1)=1	270 LET R(4,1)=1	270 LET R(4,1)=1
280 LET R(4,2)=0	280 LET R(4,2)=0	280 LET R(4,2)=0	280 LET R(4,2)=0
310 LET Z=INT(RND*9)+2	310 LET Z=INT(RND(1)*9)+2	310 LET Z=INT(RND*9)+2	310 LET Z=INT(RND(1)*9+2)
320 LET S=INT(RND*9)+2	320 LET S=INT(RND(1)*9)+2	320 LET S=INT(RND*9)+2	320 LET S=INT(RND(1)*9+2)
330 IF F$(Z,S)<>"X" THEN GOTO 310	330 IF F$(Z,S)<>"X" THEN GOTO 310	330 IF F$(Z,S)<>"X" THEN 310	330 IF F$(11*Z+S-11,11 *Z+S-11) <>"X" THEN GOTO 310
340 LET W=INT(RND*4)+1	340 LET W=INT(RND(1)*4)+1	340 LET W=INT(RND*4)+1	340 LET W=INT(RND(1)*4+1)
350 IF F$(Z+R(W,1), S+R(W,2))<>"X" THEN GOTO 310	350 IF F$(Z+R(W,1), S+R(W,2))<>"X" THEN GOTO 310	350 IF F$(Z+R(W,1), S+R(W,2))<>"X" THEN 310	350 LET X=11*(Z+R(W,1))+S +R(W,2)-11:IF F$(X,X)<>"X" THEN GOTO 310
360 IF F$(Z+2*R(W,1), S+2*R(W,2))<>" " THEN GOTO 310	360 IF F$(Z+2*R(W,1), S+2*R(W,2))<>" " THEN GOTO 310	360 IF F$(Z+2*R(W,1), S+2*R(W,2))<>" " THEN 310	360 LET X=11*(Z+2*R (W,1))+S+2*R(W,2)-11:IF F$(X,X) <>" "THEN GOTO 310
370 LET F$(Z,S)=" "	370 LET F$(Z,S)=" "	370 LET F$(Z,S)=" "	370 LET F$(11*Z+S -11,11*Z+S-11)=" ": POS.Z,S:PRINT" "
380 PRINT AT Z,S;" "	380 PRINT LEFT$(C$,Z); TAB(S);" "	380 CALL HCHAR (Z,S,32)	
390 FOR I=1 TO 10	390 FOR I=1 TO 10	390 FOR I=1 TO 10	390 FOR I=1 TO 10
400 NEXT I	400 NEXT I	400 NEXT I	400 NEXT I
410 LET F$(Z+2*R(W,1), S+2*R(W,2))="X"	410 LET F$(Z+2*R(W,1), S+2*R(W,2))="X"	410 LET F$(Z+2*R(W,1), S+2*R(W,2))="X"	410 LET X=11*(Z+2*R(W,1)) +S+2*R(W,2)-11:LET F$(X,X)="X"
420 PRINT AT Z+2*R(W,1), S+2*R(W,2);"X"	420 PRINT LEFT$(C$,Z+ 2*R(W,1));TAB(S+2* R(W,2));"X"	420 CALL HCHAR (Z+2*R(W,1), S+2*R(W,2),88)	420 POS.Z+2*R(W,1), S+2*R(W,2):PRINT"X"
430 FOR I=1 TO 20	430 FOR I=1 TO 20	430 FOR I=1 TO 20	430 FOR I=1 TO 20
440 NEXT I	440 NEXT I	440 NEXT I	440 NEXT I
450 LET F$(Z+R(W,1), S+R(W,2))=" "	450 LET F$(Z+R(W,1), S+R(W,2))=" "	450 LET F$(Z+R(W,1), S+R(W,2))=" "	450 LET X=11*(Z+R(W,1)) +S+R(W,2)-11:LET F$(X,X)=" "
460 PRINT AT Z+R(W,1), S+R(W,2);" "	460 PRINT LEFT$(C$,Z+ R(W,1)); TAB(S+R(W,2));" "	460 CALL HCHAR(Z+R(W,1), S+R(W,2),32)	460 POS.Z+R(W,1), S+R(W,2):PRINT" "
480 GOTO 310	480 GOTO 310	480 GOTO 310	480 GOTO 310

Tab. 12 Listing des Programmes Solitaire

Elementes von R hinzugezählt werden, weil dieses Einzelfeld ja vom Ausgangseinzelfeld um zwei Einzelfelder entfernt ist.

Nachdem ein Zug gefunden ist, folgt in den Programmzeilen 370 bis 460 die Ausführung des Zuges. Die Programmzeilen 390 bis 400 und 430 bis 440 sind sogenannte Warteschleifen; sie sollen nur Pausen einlegen, damit der Ablauf auf dem Bildschirm gut verfolgt werden kann. Danach erfolgt ein Rücksprung nach Programmzeile 310, und der Home-Computer sucht nach weiteren Zügen.

Der BASIC-Kurs wird im nächsten Heft fortgesetzt.

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC 20, Texas Instruments 99/4A und Atari 600 XL

In die Karten geschaut

Ob beim Pokern oder 17 und 4: der Home-Computer mischt mit und legt manchmal sogar alte Hasen und Spielratten aufs Kreuz. Dabei kann er ziemlich kühl und herzlos sein. Er hat es nicht von der Pike auf gelernt, ein Programm teilt ihm die Regeln mit und gibt ihm seine Spielstärke.

Wir wollen uns als Kiebitze betätigen und zwischen den Programmzeilen lesen. Ein einfaches Programmbeispiel für 17 und 4 soll die wesentlichen Elemente der Programmierung von Kartenspielen verständlich machen.

Das Blatt

Bei jedem Kartenspiel müssen dem Home-Computer zunächst die einzelnen Spielkarten bekannt sein. Wir wollen ein Skatblatt benutzen. Da es bei 17 und 4 nur auf den Wert der Karten ankommt und nicht auf die Farbe, genügt es, die Werte 2, 3, 4, 7, 8, 9, 10, 11 entsprechend Bube, Dame, König, Sieben, Acht, Neun, Zehn, As insgesamt viermal in einem Zahlenfeld zu speichern. Als Name des Feldes wurde im Programm K (für Karten) gewählt. Eigentlich benötigt K nur 32 Elemente. Wegen des späteren Entfernens von Karten aus dem Stapel ist es günstiger, 33 Elemente zu vereinbaren:

```
10 DIM K (33)
```

Die nächste Programmzeile beinhaltet in einer DATA-Anweisung die acht verschiedenen Werte:

```
20 DATA 2, 3, 4, 7, 8, 9, 10,
   11
```

In den Zeilen 40 bis 100 werden in zwei ineinander verschachtelten Schleifen das Feld K für die Werte der 32 Karten aufgebaut. Die Zeile 80 liest jeweils ein Element des Feldes aus der DATA-Zeile ein:

```
80 READ K (Z)
```

READ/DATA eignen sich besonders dazu, im Programm größere Datenmengen Variablen zuzuweisen.

Die Variable Z übernimmt die Aufgabe eines Zählers, der angibt, welches Element gerade an der Reihe ist. Deshalb muß Z auch bei jedem Durchlauf der inneren Schleife um Eins vergrößert werden:

```
70 LET Z=Z+1
```

In Zeile 30 wird Z zunächst auf Null gesetzt.

Die innere Schleife reicht von Zeile 60 bis Zeile 90. Sie wird von den beiden Anweisungen

```
60 FOR J=1 TO 8
90 NEXT J
```

begrenzt. Die Zeile 60 legt dabei fest, wie oft die Befehlsfolge bis zur Anweisung NEXT J durchlaufen wird. Die Laufvariable J hat beim ersten Durchlauf den Wert 1; trifft das Programm auf die Zeile 90 NEXT J, wird der Wert von J um 1 erhöht, J bekommt dann also 2 zugewiesen, und die Befehlsfolge ab Zeile 60 wird wiederholt.

Die Schleife ist abgearbeitet, wenn das Programm bei 90 NEXT J der Variablen J einen Wert über 8 zuweist. Das tritt dann ein, wenn die innere Schleife achtmal durchlaufen wurde. Daraufhin wird die nächste Zeile mit der Zeilennummer 100 bearbeitet. Die Schleife mit der Laufvariablen J ist in eine weitere Schleife mit der Laufvariablen I eingebettet:

```
40 FOR I=1 TO 4
100 NEXT I
```

Dadurch wird die innere Schleife insgesamt viermal vollkommen abgearbeitet. Die Variable Z zählt während der Ausführung immer weiter, bis sie schließlich 32 erreicht hat.

Die Anweisung READ K(Z) wird also ebenfalls 32mal ausgeführt. In der DATA-Zeile befinden sich aber nur acht Werte. Bei jedem Durchlauf der äußeren Schleife muß der Home-Computer deswegen eine Mitteilung erhalten, daß er zum Lesen wieder am Anfang der DATA-Zeile beginnen soll. Dies besorgt die Anweisung

```
50 RESTORE
```

RESTORE bewirkt, daß die READ-Anweisung wieder vom Beginn der ersten DATA-Zeile an liest.

Mit den folgenden Erklärungen wollen wir uns direkt auf das abgedruckte Listing beziehen, ohne die einzelnen Pro-

Scan: S. Höltgen

grammzeilen im Text zu wiederholen.

In den Zeilen 150 bis 170 teilt der Rechner mit, daß er einen vollständigen Stapel mit 32 Karten zur Verfügung stellt. Deshalb wird auch in Zeile 180 die Variable R (für den Rest, der später immer kleiner wird) auf 32 gesetzt.

```
10 DIM K(33)
20 DATA 2,3,4,7,8,9,10,11
30 LET Z=0
40 FOR I=1 TO 4
50 RESTORE
60 FOR J=1 TO 8
70 LET Z=Z+1
80 READ K(Z)
90 NEXT J
100 NEXT I
150 PRINT
160 PRINT "* NEUER STAPEL *"
170 PRINT
180 LET R=32
200 LET Z=0
210 LET S=0
220 LET E$=""
230 GOSUB 700
240 IF Z<=2 THEN GOTO 260
250 IF S>21 THEN GOTO 590
260 IF S>=21 THEN GOTO 550
270 IF S<16 THEN GOTO 230
280 LET C=S
300 LET Z=0
310 LET S=0
320 GOSUB 700
330 PRINT H;
340 GOSUB 700
350 PRINT "+";H;
360 IF Z<=2 THEN GOTO 380
370 IF S>21 THEN GOTO 540
380 IF S>=21 THEN GOTO 580
390 INPUT E$
400 IF E$<>"F" THEN GOTO 340
500 PRINT "= ";S
510 PRINT "DER COMPUTER HAT ";C
520 IF S>C THEN GOTO 590
530 GOTO 550
540 PRINT "= ";S
550 PRINT "VERLOREN"
560 PRINT
570 GOTO 200
580 PRINT "= ";S
590 PRINT "GEWONNEN"
600 PRINT
610 GOTO 200
700 LET Z=Z+1
710 LET A=INT(RND(1)*R+1)
720 LET S=S+K(A)
730 LET R=R-1
740 LET H=K(A)
750 FOR I=A TO R
760 LET K(I)=K(I+1)
770 NEXT I
780 IF R=0 THEN GOTO 30
790 RETURN
```

Als erster zieht der Computer; zur Vorbereitung werden die Zahl Z der genommenen Karten und die Augensumme S auf Null gesetzt. In Zeile 230 soll zum erstenmal eine Karte genommen werden. Dies kommt auch an mehreren anderen Stellen des Programms vor. Deshalb kann man für diese Sonderaufgabe ein kleines Extra-Programm schreiben und spart sich dadurch, die notwendige Befehlsfolge im Programm mehrmals wiederholen zu müssen. Man nennt solche Programme Unterprogramme; sie werden mit GOSUB aufgerufen.

GOSUB ruft ein Unterprogramm auf.

Karten nehmen

Das Unterprogramm zum Nehmen einer Karte steht von Zeile 700 bis 790. In Zeile 700 wird die Zahl Z der genommenen Karten um eins erhöht. Zeile 710 bestimmt, die wievielte Karte aus dem Stapel genommen wird.

Vielleicht haben Sie sich schon gewundert, daß die Karten zu Beginn gar nicht gemischt wurden. Diese Arbeit macht sich der Home-Computer nicht; es wird statt dessen aus dem geordneten Stapel an zufälliger Stelle eine Karte herausgezogen, was auf das gleiche hinausläuft.

Der Rest-Stapel

Nachdem die Stelle A innerhalb des Stapels bestimmt wurde, wird der Wert der A-ten-Karte in Zeile 720 zur bisherigen Augensumme hinzuaddiert. Es befindet sich jetzt eine Karte weniger im Stapel; das wird in Zeile 730 berücksichtigt.

Die gezogene Karte muß jetzt aus dem Stapel entfernt werden. Nach dem Löschvorgang weiß das Programm den Wert der genommenen Karte nicht mehr. Deshalb ist die Zeile 740 nötig, die diesen Wert in der Hilfsvariablen H aufbewahrt. Die eigentliche Löschung wird in den Zeilen 750 bis 770 vorgenommen. Sie geschieht dadurch, daß im Feld K ab dem A-ten-Wert alle Werte durch den jeweils nächsten überschrieben werden. Zeile 780 veranlaßt einen Neubeginn mit einem vollständigen Stapel, wenn alle Karten aufgebraucht sind. Am Ende des Unterprogramms bewirkt die Anweisung RETURN, einen Rücksprung.

RETURN springt aus einem Unterprogramm heraus, und das Programm wird hinter der Zeile fortgesetzt, in der das Unterprogramm aufgerufen wurde.

Mit der Analyse befinden wir uns jetzt in der Programmzeile 240. Diese und die nächsten zwei Zeilen sorgen dafür, daß der Computer sich geschlagen gibt, wenn er mehr als 21 hat. Natürlich ist hier auch der Sonderfall berücksichtigt, daß man mit zwei Assen gewonnen hat. Die Meldungen „GEWONNEN" und „VERLOREN" in den Zeilen 550 und 590 beziehen sich immer auf den menschlichen Partner des Home-Computers.

Die Zeile 270 verrät die simple Strategie des Home-Computers: Er hört auf, Karten zu nehmen, wenn seine Augenzahl 16 oder mehr ist. Zeile 280 übergibt die Augensumme des Computers an die Variable C, da S jetzt für sein Gegenüber gebraucht wird. Jedesmal, wenn dieser gezogen hat, wird der gezogene Wert auf dem Bildschirm angezeigt. Er zieht mindestens zweimal, wie die Zeilen 320 und 340 zeigen; danach nur noch, wenn bei der folgenden Eingabe nicht „F" (für fertig) eingegeben wird. Die Zeilen 360 bis 380, die Verlust oder Gewinn kalkulieren, entsprechen den Zeilen 240 bis 260 beim Home-Computer, nur hat natürlich der Home-Computer genau dann gewonnen, wenn sein Gegner verloren hat und umgekehrt.

Gewinn – Verlust

Die Zeilen 500 bis 530 behandeln den Normalfall, das heißt, keiner hat 21 erreicht und keiner hat überzogen. Zum Gewinn muß der Home-Computer mindestens so viel Augen wie sein Gegenüber haben.

In der abgedruckten Version läuft das Programm auf einem VC 20. Für andere Home-Computer können Änderungen bezüglich der Stringvariablen E$, der IF-THEN-Befehle und der RND-Funktion nötig sein. *Der BASIC-Kurs wird im nächsten Heft fortgesetzt.*

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC 20, Texas Instruments 99/4A und Atari 600 XL

Wie funktionieren die Funktionen?

In den bisherigen Folgen des BASIC-Kurses hatten wir bereits mehrere Funktionen kennengelernt. Im „Galgenmännchen"-Programm aus HC-Januar 1984 diente die Funktion LEN dazu, die Länge des Rätselwortes zu bestimmen. LEFT$ und RIGHT$ wurden im gleichen Programm dazu benutzt, um den linken und den rechten Teil des Rätselwortes abzugreifen.

Die Funktion RND sorgte in HC-Dezember 1983 dafür, daß beim Schatzsuche-Programm der Schatz gut versteckt wurde; der Home-Computer wählte dazu innerhalb eines Feldes zufällig eine Stelle aus.

Beim Primzahl-Programm in HC-Februar 1984 kam die Funktion INT zum Einsatz, die eine Zahl zur nächstkleineren ganzen Zahl abrundet.

Es gibt noch eine ganze Reihe anderer Funktionen, die von Home-Computern zur Verfügung gestellt werden. Die verschiedenen Home-Computer-Modelle sind leider nicht alle gleich gut mit Funktionen ausgestattet.

Einen ganzen Bereich von Funktionen, nämlich die trigonometrischen, haben wir bisher noch gar nicht behandelt. Zu ihnen gehören SIN, COS, TAN, ASN, ACS, ATN. Sie werden zum Beispiel gebraucht, wenn man Kreise zeichnet oder Körper perspektivisch darstellen will.

Die meisten Funktionen benötigen ein Argument. Dies sind Zahlen, Texte oder entsprechende Variable oder Ausdrücke, mit denen sie rechnen sollen. Das Argument steht meist in Klammern hinter dem Funktionsnamen. Man kann zwischen Funktionen unterscheiden, die als Ergebnis eine Zahl liefern, und Funktionen, die als Ergebnis einen Text liefern. Funktionen, deren Ergebnis ein Text ist, haben am Ende ihres Namens das Zeichen $; wenn eine Funktion eine Zahl liefert, endet ihr Name also ohne $.

Aus der abgebildeten Tabelle geht für vier Home-Computer hervor, wie sie mit Funktionen ausgestattet sind. Das Zeichen + bedeutet, daß die Funktion vorhanden ist; das Zeichen – bedeutet: nicht vorhanden. Falls möglich, werden bei fehlenden Funktionen Hinweise gegeben, welche andere Funktion der fehlenden entspricht oder wie man sonst die fehlende ersetzen kann.

Mit in die Tabelle aufgenommen wurden die logischen Operatoren AND, OR, NOT. Durch sie können Bedingungen verknüpft werden, so daß zum Beispiel ein Befehl nur dann ausgeführt wird, wenn von zwei Bedingungen beide erfüllt sind (AND-Operator). Soll es zur Befehlsausführung schon ausreichen, daß eine von zwei Bedingungen erfüllt ist, so benutzt man den OR-Operator. NOT dient zur Umkehrung einer Bedingung.

Beim TI-99/4A fällt auf, daß er in der Grundversion keine logischen Operatoren kennt. Dort müssen sie beim Programmaufbau durch Verwendung mehrerer Programmzeilen nachgeahmt werden.

In der Tabelle stehen T$ und U$ für zwei beliebige Texte, Textvariable oder Textausdrücke und Z, Z1, Z2 für zwei beliebige Zahlen, Zahlenvariable oder Zahlausdrücke.

Der BASIC-Kurs wird im nächsten Heft fortgesetzt.

	Spectrum	VC 20	TI-99/4A	Atari 600 XL	Argument	Ergebnis	Bedeutung	Beispiel
ABS (Z)	+	+	+	+	Zahl	Zahl	Absolutbetrag	ABS(4)=4; ABS(–13.4)=13.4
ACS (Z)	+	–	–	–	Zahl	Zahl	Arcuscosinus im Bogenmaß	ACS(0)=PI/2
AND	+	+	–	+	2 Bedingungen	Bedingung	logisches „und"	IF A=D AND B=C THEN GOTO 100
ASC (T$)	→CODE	+	+	+	Text	Zahl	Umwandlung: Zeichen in (ASCII-)Code-Zahl	ASC("A")=65
ASN (Z)	+	–	–	–	Zahl	Zahl	Arcussinus im Bogenmaß	ASN(1)=PI/2
ATN (Z)	+	+	+	+	Zahl	Zahl	Arcustangens im Bogenmaß	ATN(1)=PI/4
CHR$ (Z)	+	+	+	+	Zahl	Text	Umwandlung: Code-Zahl in Zeichen (Charakter)	CHR$(49)="1"
CODE (T$)	+	→ASC	→ASC	→ASC	Text	Zahl	Umwandlung: Zeichen in Code-Zahl	CODE("1")=49

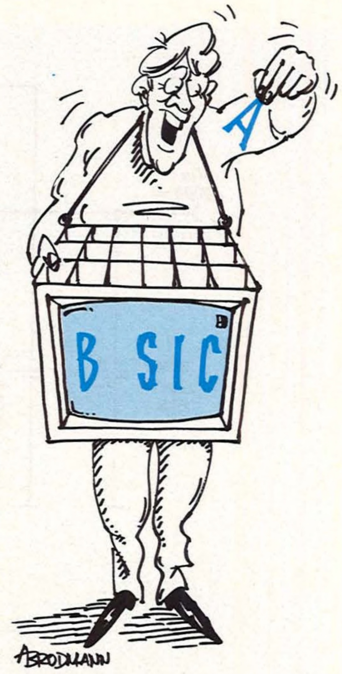
Scan: S. Höltgen

	Spectrum	VC 20	TI-99/4A	Atari 600 XL	Argument	Ergebnis	Bedeutung	Beispiel
COS (Z)	+	+	+	+	Zahl	Zahl	Cosinus eines Winkels im Bogenmaß	COS(PI)=1
EXP (Z)	+	+	+	+	Zahl	Zahl	Exponentialfunktion (e hoch)	EXP(1)=e
FN XX	+	+	–	–	(Zahl)	Zahl	selbstdefinierte Funktion	FN KS=SQR(X*X+Y*Y)
FRE (0)	–	+	–	+	Zahl	Zahl	freier Speicherplatz in Byte	–
INKEY$	+	→Befehl GET	→CALL KEY	→Befehl GET	–	Text	Tastaturabfrage weist Zeichen zu	10 IF INKEY$="" THEN GOTO 10
INT (Z)	+	+	+	+	Zahl	Zahl	nächstkleinere ganze Zahl	INT(2.6)=2; INT(−11.18)=−12; INT(4)=4
LEFT$ (T$,Z)	=T$ (T0 Z)	+	=SEG$ (T$,1,Z)	=T$(1,Z)	Text, Zahl	Text	greift linken Teil eines Textes heraus	LEFT$("FALLE",2)="FA"
LEN (T$)	+	+	+	+	Text	Zahl	Länge eines Textes	LEN("HOMECOMPUTER")=12
LN (Z)	+	=LOG(Z)	=LOG(Z)	–	Zahl	Zahl	Logarithmus zur Basis e	LN(E↑4)=4
LOG (Z)	–	–	–	=CLOG (Z)	Zahl	Zahl	Logarithmus zur Basis 10	LOG(1000)=3
MID$ (T$,Z1,Z2)	=T$ (Z1T0,Z1 +Z2−1)	+	=SEG$ (T$,Z1, Z2)	=T$(Z1, Z1+Z2−1)	Text, Zahl, Zahl	Text	greift Teil eines Textes heraus	MID$("TELOFON",4,3)= "OFO"
NOT	+	+	–	+	1 Bedingung	Bedingung	logische Verneinung	IF NOT A<0 THEN RETURN
OR	+	+	–	+	2 Bedingungen	Bedingung	logisches „oder“	IF A<0 OR A>9 THEN GOTO 90
PEEK (Z)	+	+	–	+	Zahl	Zahl	Wert einer einzelnen Speicherzelle	Z=PEEK(32 000)
PI	+	–	–	–	–	Zahl	3.14159...	–
POS (Z)	–	+	–	–	Zahl	Zahl	Spaltenposition des Cursors	–
POS (T$,U$,Z)	–	–	+	–	Text, Text, Zahl	Zahl	Position des ersten Zeichens von U$ in T$	POS("KARNEVAL", "NEBEN",2)=4
RIGHT$ (T$,Z)	=T$ (LENT$ −Z+1 T0 LENT$)	+	=SEG$ (LEN(T$) −Z+1,Z)	=T$ (LEN(T$) −Z+1,LEN (T$))	Text, Zahl	Text	greift rechten Teil eines Textes heraus	RIGHT$("SINFONIE",3) ="NIE"
RND (Z)	RND	RND (Z)	RND	RND (Z)	(Zahl)	Zahl	Zufallszahl zwischen 0 und 1 (mit 0, ohne 1)	RND=0.32579
SEG$ (T$,Z1,Z2)	=T$(Z1 Z2−1)	=MID$ (T$, Z1,Z2)	+	=T$(Z1,Z1 +Z2−1)	Text	Text	greift Teil eines Textes heraus	SEG$("LASTWAGEN",2,3) ="AST"
SGN (Z)	+	+	+	+	Zahl	Zahl	Vorzeichen (Signum) einer Zahl (1 oder −1)	SGN(−3)=−1
SIN (Z)	+	+	+	+	Zahl	Zahl	Sinus eines Winkels im Bogenmaß	SIN(PI/2)=1
SPC (Z)	–	+	–	–	Zahl	–	Vorschub der Ausgabe um Z Spalten	–
SQR (Z)	+	+	+	+	Zahl	Zahl	Wurzel (squareroute) einer Zahl	SQR(49)=7
STR$ (Z)	+	+	+	+	Zahl	Text	Umwandlung Zahl in Text (string)	STR$(17)="17"
TAB (Z)	+	+	–	–	Zahl	–	Ausgabe ab angegebener Spalte Z	PRINT TAB(10);"SPIELER"
TAN (Z)	+	+	+	–	Zahl	Zahl	Tangens eines Winkels im Bogenmaß	TAN(PI/4)=1
USR (Z)	+	+	–	+	Zahl	Zahl	Aufruf eines Programmes in Maschinensprache	I=USR(30 000)
VAL (T$)	+	+	+	–	Text	Zahl	Umwandlung Text in Zahl (value)	VAL("93")=93

Die wichtigsten Funktionen und logischen Operatoren der Home-Computer

Scan: S. Höltgen

Programmieren in BASIC



In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC 20, Texas Instruments 99/4A und Atari 600 XL

Kenntnisse über die Art der Speicheraufteilung eines Home-Computers können wichtige Anregungen für die Gestaltung von eigenen Programmen liefern.

Da wäre zunächst einmal der ROM-Speicher (Read-Only-Memory). Er kann vom Programmierer nicht beeinflußt werden, und das ist gut so, denn er enthält zum Beispiel den BASIC-Interpreter, der zu jeder Zeit in tatsächlich unveränderter Form zur Verfügung stehen muß.

Anders verhält es sich mit dem RAM-Speicher (Random-Access-Memory): Seine Speicherzellen können ihren Inhalt nach Belieben wechseln. Bedeutende Bestandteile des RAM-Speichers sind zum Beispiel das BASIC-Programm und die Variablen.

Um einzelne Speicherzellen auf ihren Inhalt hin testen zu können, gibt es die bekannte Funktion PEEK.

LET A=PEEK(20500) weist zum Beispiel der Variablen A den Inhalt der Speicherzelle 20500 zu, verschlüsselt als Zahl zwischen 0 und 255. In jeder Speicherzelle ist nämlich Platz für den Informationsgehalt von einem Byte, und dieser Informationsgehalt kann gerade 256 verschiedene Zustände annehmen.

Das Gegenstück zur Funktion PEEK ist der Befehl POKE. Mit ihm kann man einzelne Speicherstellen verändern.

POKE 14381,22 zum Beispiel schreibt in die Speicherzelle mit der Nummer 14381 das Byte, dessen dezimale Darstellung 22 ist. Wendet man POKE auf eine Speicherstelle innerhalb des ROM an, so hat er keine Wirkung.

Ein zielloses Verändern irgendwelcher Speicherstellen mit POKE kann ungewollte Folgen haben, wenn zum Beispiel dadurch innerhalb des BASIC-Programmes sinnlose Bytefolgen entstehen.

Bildschirmspeicher

Besonders nützlich und augenfällig ist die Verwendung von PEEK und POKE für den Bildschirmspeicher: Ihre Arbeitsweise bzw. ihre Wirkung ist unmittelbar ersichtlich. Bei den meisten Home-Computern beginnt der Bildschirmspeicher an einer festen Stelle im RAM. Die Größe des Bereichs ergibt sich aus der Anzahl der Bildschirmzeilen und -spalten. So reicht der Bildschirmspeicher zum Beispiel beim VC 20 von den Adressen 7680 bis 8185 und beim C 64 von 1024 bis 2023.

Eine große Ausnahme bildet hier der ZX81. Der Bildschirmspeicher liegt hier hinter dem BASIC-Programm, und sein Anfang variiert je nach Länge des BASIC-Programmes (in der 1K-Version variiert aus Ersparnisgründen auch die Länge des Bildschirmspeichers). An welcher Speicherstelle das erste Zeichen des Bildschirmspeichers momentan steht, läßt sich dort mit Hilfe der PEEK-Funktion erfragen:

PRINT PEEK 16396 + 256 * PEEK 16397 +1

Im Bildschirmspeicher der meisten Home-Computer-Modelle stehen nacheinander zunächst alle Zeichen der ersten Bildschirm-Zeile, dann die der zweiten Bildschirmzeile usw. in der dem Zeichen zugeordneten Codezahl. Das gilt nicht für den Spectrum, bei dem die Zeilen in einer anderen Folge aneinandergereiht sind und der dank seiner hochauflösenden Grafik jedes Zeichen gemäß dessen Struktur noch einmal in 8 Byte zerpflückt. Der ZX81 fügt nach jeder Zeile ein 33.) Byte hinzu (Codezahl für "NEW LINE").

Um bestimmte Zeichen vom Bildschirm zu lesen, muß man also die jeweilige Lage und Struktur des Bildschirmspeichers berücksichtigen. Programmiert man in Maschinensprache, muß man in jedem Fall so vorgehen. Beim ZX81 und Commodore führt auch in BASIC kein Weg daran vorbei. Die meisten anderen Home-Computer stellen jedoch eine BASIC-Funktion zur Verfügung, die das Lesen vom Bildschirm aus dem Programm heraus sehr erleichtert

(Tab. 13). Die Listings von Tabelle 14 zeigen eine Anwendung. Der Home-Computer zeichnet dort ausgehend von einer zufällig gewählten Stelle innerhalb eines vorgegebenen Rahmens Irrwege auf, das heißt er wählt nach jedem Schritt eine zufällige Richtung. Dabei betritt er nie eine Stelle, an der er schon einmal gewesen ist. Dabei „verfranzt" er sich natürlich irgendwann, und das Programm muß abgebrochen werden.

Die Abfrage „Bin ich dort schon gewesen?" ist nun genau ein Beispiel für das Lesen vom Bildschirm (Zeile 640 bzw. 640 und 650).

Die Auswahl einer zufälligen Richtung spielte innerhalb des BASIC-Kurses schon einmal eine Rolle, nämlich beim Programm „Solitaire" aus dem März-Heft von HC. Im Programm „Irrweg" wird hier in den Zeilen 600 bis 640 eine vollkommen andere Methode benutzt, die den gleichen Zweck erfüllt.

Als kleine Aufgabe versuchen Sie doch bitte, diese Zeilen zu analysieren.

Der BASIC-Kurs wird im nächsten Heft fortgesetzt.

ZX 81	Spectrum	VC 20	C 64	TI-99/4A	Atari
PEEK (PEEK 16396 + 256*PEEK 16397 + 1 + 33*Z + S) ergibt Code-Zahl des Zeichens	SCREEN$(Z,S) ergibt Zeichen	PEEK (7680 + 22*Z + S) ergibt Code-Zahl des Zeichens	PEEK (1024 + 40*Z + S) ergibt Code-Zahl des Zeichens	CALL GCHAR (Z,S,A) Code-Zahl des Zeichens befindet sich in A	LOCATE S,Z,A Code-Zahl des Zeichens befindet sich in A

Tab. 13 Lesen eines Zeichens aus Zeile Z und Spalte S des Bildschirms

Spectrum	VC 20	TI-99/4A	Atari
	5 LET C$="SQQQQQQQQQQQQQQQQQQQQ"	5 CALL CHAR(33,"F F F F F F F F F F F F F F F F")	5 GRAPHICS 3+16
10 INPUT X	10 INPUT X	10 INPUT X	10 INPUT X
20 INPUT y	20 INPUT Y	20 INPUT Y	20 INPUT Y
300 FOR I=2 TO 2*(X+1)	300 FOR I=2 TO 2*(X+1)	300 FOR I=2 TO 2*(X+1)	300 FOR I=3 TO 2*(X+1)+1
310 PRINT AT 2, i; CHR$ (43)	310 PRINT LEFT$(C$,2); TAB(I);CHR$(113)	310 CALL HCHAR(2,I,33)	310 POSITION I,2:PRINT CHR$(160)
320 PRINT AT 2*(Y+1) ,i;CHR$ (43)	320 PRINT LEFT$ (C$,2*(Y+1)); TAB(I);CHR$(113)	320 CALL HCHAR(2*(Y+1),I,33)	320 POSITION I, 2*(Y+1):PRINT CHR$(160)
330 NEXT i	330 NEXT I	330 NEXT I	330 NEXT I
340 FOR i=2 TO 2*(Y+1)	340 FOR I=2 TO 2*(Y+1)	340 FOR I=2 TO 2*(Y+1)	340 FOR I=2 TO 2*(Y+1)
350 PRINT AT i, 2; CHR$ (43)	350 PRINT LEFT$(C$,I); TAB(2);CHR$(113)	350 CALL HCHAR(I,2,33)	350 POSITION 3,I:PRINT CHR$(160)
360 PRINT AT i, 2*(X+1);CHR$ (43)	360 PRINT LEFT$ (C$,I);TAB (2*(X+1));CHR$ (113)	360 CALL HCHAR(I,2*(X+1),33)	360 POSITION 2*(X+1)+1, I:PRINT CHR$(160)
370 NEXT i	370 NEXT I	370 NEXT I	370 NEXT I
500 LET XX=(INT (RND* (X−1))+2)*2	500 LET XX=(INT(RND (1)*(X−1))+2)*2	500 LET XX=(INT(RND* (X−1))+2)*2	500 LET XX=(INT(RND(1) *(X−1)) +2)*2+1
510 LET YY=(INT (RND*(Y−1))+2)*2	510 LET YY=(INT (RND(1)*(Y−1))+2)*2	510 LET YY=(INT (RND*(Y−1))+2)*2	510 LET YY=(INT (RND(1)*(Y−1))+2)*2
600 LET Z1=INT (RND*2)	600 LET Z1=INT(RND(1)*2)	600 LET Z1=INT(RND*2)	600 LET Z1=INT(RND(1)*2)
610 LET Z2=INT (RND*2)*2−1	610 LET Z2=INT(RND(1) *2)*2−1	610 LET Z2=INT(RND*2)*2−1	610 LET Z2=INT(RND(1) *2)*2−1
620 LET XA=XX+Z1*Z2*2	620 LET XA=XX+Z1*Z2*2	620 LET XA=XX+Z1*Z2*2	620 LET XA=XX+Z1*Z2*2
630 LET YA=YY+(1−Z1)*Z2*2	630 LET YA=YY+(1−Z1)*Z2*2	630 LET YA=YY+(1−Z1)*Z2*2	630 LET YA=YY+(1−Z1)*Z2*2
640 IF SCREEN$ (YA,XA) =CHR$ (43) THEN GO TO 600	640 IF PEEK (7680+22* (YA−1)+XA)=81 THEN 600	640 CALL GCHAR(YA,XA,A)	640 LOCATE XA,YA,A
		650 IF A=33 THEN 600	650 IF A=160 THEN GOTO 600
700 PRINT AT YY,XX;CHR$ (43)	700 PRINT LEFT$(C$,YY); TAB(XX);CHR$(113)	700 CALL HCHAR(YY,XX,33)	700 POSITION XX,YY:PRINT CHR$(160)
710 PRINT AT (YY+YA)/2, (XX+XA)/2;CHR$ (43)	710 PRINT LEFT$ (C$,(YY+YA)/2); TAB((XX+XA)/2); CHR$(113)	710 CALL HCHAR((YY+YA)/2, (XX+XA)/2,33)	710 POSITION (XX+XA)/2, (YY+YA)/2:PRINT CHR$ (160)
720 LET XX=XA	720 LET XX=XA	720 LET XX=XA	720 LET XX=XA
730 LET YY=YA	730 LET YY=YA	730 LET YY=YA	730 LET YY=YA
740 GOTO 600	740 GOTO 600	740 GOTO 600	740 GOTO 600
Änderungen für ZX 81 640 IF PEEK (PEEK 16396 +256*PEEK 16397+1 +33*YA+XA) =43 THEN GOTO 600	**Änderungen für C 64** 640 IF PEEK(1024+40* (YA−1)+XA)=81 THEN 600		

Tab. 14 Listings von „Irrweg"

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore VC 20, Texas Instruments 99/4A und Atari 600 XL

Nach dem Einschalten des Home-Computers kann man ihm kein X für ein U vormachen: Er kennt sofort das Alphabet, weiß, wie die zehn Ziffern und die Satzzeichen aussehen und erinnert sich sofort an seine Grafik- und Sonderzeichen.

Da man sie ihm nicht jedesmal neu vermittelt und sie ihm auch nicht aus heiterem Himmel eingefallen sein können, müssen alle Zeichen im ROM-Speicher fest verankert sein: Er ist also auch im ausgeschalteten Zustand kein Analphabet.

Im Zeichen- oder Charaktergenerator genannten Speicherbereich sind die Zeichen bei den meisten Home-Computern (nicht beim Atari) in der Reihenfolge der Code-Tabelle angeordnet.

Jedem Zeichen steht für seine Struktur eine 8∗8-Matrix zur Verfügung. Wie sich ein Zeichen in dieser Matrix zusammensetzt, soll an einem Beispiel (Tabelle 15) gezeigt werden, das in keinem Zeichenvorrat eines Home-Computers vorkommt, jedem Leser aber wohlvertraut ist.

Wenn Zahlen das System wechseln

Zur Beschreibung einer Zeile der Matrix kann man verschiedene Zahlensysteme wählen. Dem Erscheinungsbild am nächsten kommt das duale oder binäre Zahlensystem, bei dem zum Beispiel die obere Zeile folgendermaßen aussieht: 10010000. Eine Eins gibt jeweils an, daß das entsprechende Kästchen gefüllt ist; eine Null steht für ein leeres Kästchen.

In der Sprache des dualen Zahlensystems haben wir es mit 8 bit zu tun, von denen nur das bit 7 und das bit 4 „gesetzt" sind. 8 bit bilden zusammen ein Byte. Weil die Matrix aus acht Zeilen besteht und in einer Speicherstelle genau ein Byte Platz hat, benötigt ein Zeichen also acht Speicherstellen im Charaktergenerator.

Wenn man eine Speicherstelle jedoch mit der Funktion PEEK abfragt, findet man seinen Inhalt stets als Dezimalzahl vor. Welche Dezimalzahl entspricht nun der Dualzahl 10010000? Die einzelnen bit der Dualzahl sind von rechts nach links mit 0 bis 7 durchnumeriert. Man muß jetzt nur Potenzen von 2 bilden mit den Nummern der gesetzten bit als Exponenten und die Ergebnisse addieren:

$2^7 + 2^4$
$= 128 + 16 = 144$

Für Zeile 4 des Zeichens ergibt sich als weiteres Beispiel dann:

$2^7 + 2^4 + 2^3 + 2^2$
$= 128 + 16 + 8 + 4$
$= 156$

Das Zeichen aus Tabelle 16 ist in keinem Zeichenvorrat enthalten. Man kann jedoch mit fast allen Home-Computern zusätzlich eigene Zeichen definieren. Die Methoden weichen von Home-Computer zu Home-Computer etwas voneinander ab.

Reservierte Plätze

Grundsätzlich kann man zwei Methoden unterscheiden: Es sind in der Code-Tabelle von vornherein Plätze reserviert, die zur Aufnahme eigener Zeichen gedacht sind; oder zur Verwendung eigener Zeichen muß zunächst für den gesamten Zeichengenerator ein ganz neuer Speicherbereich gewählt werden.

Der Spectrum und der TI-99/4A gehören zur Gruppe der Rechner, die die erste Methode benutzen. Beim Spectrum können die Zeichen mit den Code-Zahlen 144 bis 164 selbst definiert werden; solange sie nicht anders aufgefüllt werden, steht dort eine Teilkopie des Alphabets von A bis U. Tabelle 16 zeigt, wie man mit dem Spectrum und drei anderen Home-Computern das Zeichen aus Tabelle 15 erzeugen und drucken kann.

CHR$(144) im Spectrum-Programm kann man auch durch „A" ersetzen. Dabei muß vor Eingabe des Buchstabens A in den GRAPHICS-Mode gewechselt werden. Ein auf diese Weise eingegebenes A läßt sich aber im Listing nicht von einem normalen A unter-

Scan: S. Höltgen

scheiden, was zu durch Doppeldeutigkeit hervorgerufenen Schwierigkeiten beim Eintippen von Spectrum-Programmen führen kann.

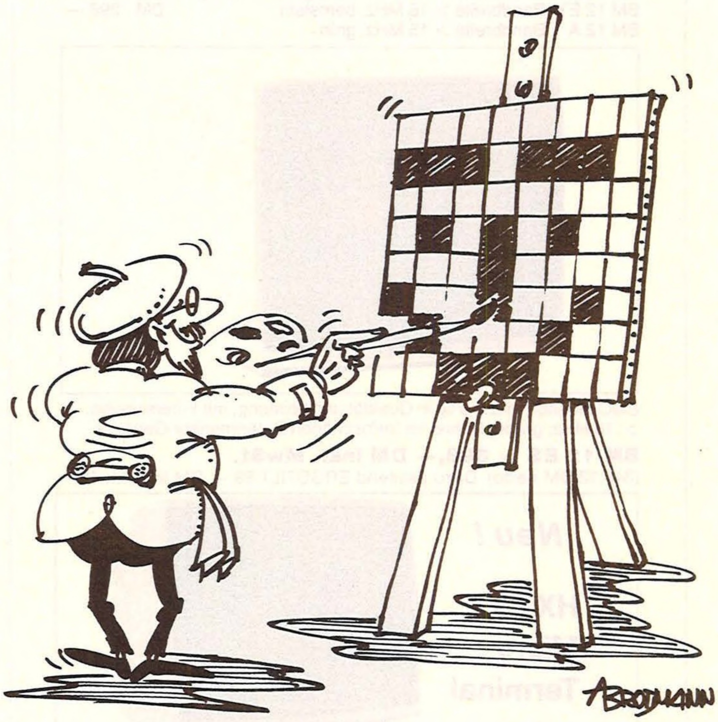
Der TI-99/4A benutzt hier nicht das dezimale Zahlensystem, sondern das hexadezimale Zahlensystem. Zur Darstellung eines Byte in hexadezimaler Form stellt man sich am besten die 8 bit aufgeteilt in zwei Gruppen zu jeweils nur 4 bit vor. So führt zum Beispiel beim Zeichen aus Tabelle 15 die erste Vierergruppe der Zeile 3 zur Dezimalzahl 9, die zweite Vierergruppe führt zur Dezimalzahl 12.

Im hexadezimalen Zahlensystem ersetzt man nun die Dezimalzahlen ab 10 durch die ersten Buchstaben des Alphabets. 10 wird also zu A und 12 zu C. Der letzte verwendete Buchstabe ist F, denn mit 4 bit kann maximal die Dezimalzahl 15 dargestellt werden. Die Zeile 3 aus Tabelle 15 lautet also 9 C in hexadezimaler Form.

Ein Zeiger muß verbogen werden

Beim TI-99/4A sind die Zeichen mit den Code-Zahlen 128 bis 159 für eigene Definitionen gedacht. Es können aber auch alle anderen schon vorhandenen Zeichen mit den Code-Zahlen 32 bis 127 umdefiniert werden.

Zur Gruppe von Rechnern, bei denen es zur Definition nur eines einzigen neuen Zeichens erforderlich ist, einen ganz neuen Speicherbereich als Zeichengenerator zu benutzen, gehören der VC 20 und der Atari 600 XL.



Zeilen	128	64	32	16	8	4	2	1	Dezimal	Hexadezimal
0	■			■					144	90
1	■			■					144	90
2	■	■	■	■					240	F0
3	■			■	■	■			156	9C
4	■		■	■					176	B0
5			■						32	20
6			■						32	20
7				■	■	■			28	1C
Bit	7	6	5	4	3	2	1	0		

Tab. 15: Der Aufbau eines selbstdefinierten Zeichens in verschiedenen Zahlensystemen

Durch Beeinflussung von Systemvariablen wird dort der Zeiger, der zunächst auf den Original-Zeichensatz eingestellt ist, auf einen freien Speicherplatz „verbogen". Dies hat zur Folge, daß der gesamte Original-Zeichensatz nicht mehr erreichbar ist und erst in den freien Speicherplatz kopiert werden müßte, wenn er zu großen Teilen weiterverwendet werden soll.

Beim VC 20 läßt
POKE 36869,253
den Zeichengenerator an der Speicherstelle 5120 beginnen. Beim Atari 600 XL wird durch
LET Z = PEEK(742)−4
POKE 756,Z
der Zeiger auf 1024 Byte unterhalb der höchsten zur Verfügung stehenden Speicheradresse eingestellt (in der Grafikbetriebsart 0 muß die Anfangsadresse des Zeichengenerators beim Atari 600 XL immer durch 1024 teilbar sein).

Dem Spectrum-Besitzer steht außer den 21 zur Eigendefinition reservierten Zeichen, wenn diese nicht ausreichen, auch die Methode zur Verfügung, die die Rechner der zweiten Gruppe auszeichnet: Durch Verändern der Systemvariable in den Speicherstellen 23606 und 23607 kann auch hier die Anfangsadresse des Zeichensatzes neu festgelegt und dieser komplett neu definiert werden.

Der BASIC-Kurs wird im nächsten Heft fortgesetzt.

Spectrum	VC 20	TI-99/4A	Atari 600 XL
	10 POKE 36869,253		10 GRAPHICS 0 20 LET Z=PEEK(742)−4 30 POKE 756,Z
50 FOR I=0 TO 7 60 READ B 70 POKE USR CHR$ 144+I,B	50 FOR I=0 TO 7 60 READ B 70 POKE 5120+I,B	70 CALL CHAR(128,"9090F09C B020201C")	50 FOR I=0 TO 7 60 READ B 70 POKE Z*256+I,B
80 NEXT I 90 DATA 144,144,240,156,176, 32,32,28	80 NEXT I 90 DATA 144,144,240,156,176, 32,32,28		80 NEXT I 90 DATA 144,144,240,156,176, 32,32,28
100 FOR I=1 TO 440 110 PRINT CHR$ 144; 120 NEXT I	100 FOR I=1 TO 440 110 PRINT "@"; 120 NEXT I	100 FOR I=1 TO 20 110 FOR J=1 TO 32 120 CALL HCHAR(I,J,128) 130 NEXT J 140 NEXT I	100 FOR I=1 TO 440 110 PRINT " "; 120 NEXT I

Tab. 16: Programme zum Erzeugen und Printen eines selbstdefinierten Zeichens

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore Texas Instruments 99/4A und Atari 600 XL

Um Objekte auf dem Bildschirm zu bewegen, kann man so vorgehen: Man wählt ein geeignetes Zeichen aus dem Zeichenvorrat oder definiert sich ein eigenes Zeichen, das dem Objekt im Aussehen recht nahe kommt; man schreibt das Zeichen gezielt an eine Stelle des Bildschirms, schreibt das gleiche Zeichen noch einmal in die nähere Umgebung und löscht das Zeichen an der alten Stelle durch Überschreiben mit einem Leerzeichen usw. Dadurch entsteht der Eindruck von Bewegung.

Sprites machen's möglich

Die Nachteile dieser Methode sind vor allem, daß ein einzelnes Zeichen nicht ausreicht, um größere Objekte darzustellen, daß die Bewegung zu langsam und zu ruckartig verläuft, bestimmte Richtungen nur im Zickzackkurs eingeschlagen werden können, der Programmierer bei Bewegungen vor einem zuvor eigens aufgebauten Hintergrund die beschädigten Stellen stets wieder restaurieren muß.

Einige Home-Computer stellen deswegen in ihrem Betriebssystem ein Instrument zur Verfügung, das die genannten Schwierigkeiten vergessen macht: das Sprite.

Darüber hinaus bieten die Sprites noch weiteren nützlichen Service. So lassen sie sich zum Beispiel leicht vergrößern. Man kann Kollisionen zwischen Sprites bequem abfragen. Zwischen den einzelnen Sprites und bezüglich des Hintergrunds lassen sich Rangfolgen festlegen, die beim Zusammentreffen bestimmen, welches Objekt welches andere verdeckt; dadurch können relativ einfach räumliche Effekte erzielt werden.

Leider kennen nicht alle Home-Computer Sprites. Zu den in dieser Hinsicht benachteiligten Rechnern gehören der ZX81, der Spectrum und der VC 20 im Gegensatz zum Commodore 64, dem TI-99/4A (jedoch nur mit Extended-BASIC) und den Atari-Rechnern (dort Player genannt). Die Benutzung der Sprites ist auf diesen drei Rechnern jedoch recht unterschiedlich. In dieser Ausgabe von HC ist unter dem Namen „Spectrum-Sprites" ein Programm-Listing abgedruckt und beschrieben, das für den Spectrum die Lücke teilweise zu schließen vermag.

Durch Beschränkung auf die wichtigsten Details und die im BASIC-Kurs übliche Gegenüberstellung der Home-Computer soll verdeutlicht werden, worauf es bei den Sprites im wesentlichen ankommt.

Der C 64 hat die größten

Der C 64 ist für die Behandlung von insgesamt acht Sprites ausgelegt. Für jedes Sprite steht eine 24∗21-bit-Matrix zur Verfügung. Es sind also 63 Byte nötig, um ein ganzes Sprite zu definieren. Zum Aufbau kommen uns die Überlegungen aus der vorangegangenen Folge des BASIC-Kurses zugute. Die einzelnen Byte werden als Dezimalzahlen in DATA-Zeilen aufgeführt, von dort gelesen und in einen geeigneten Speicher-Bereich gepoket.

In den Zeilen 250 bis 310 des Commodore-Listings aus Tabelle 18 ist auf diese Weise ein Ballon versteckt, und in den Zeilen 360 bis 420 eine Wolke. Für den Ballon wurde der Bereich von Speicherstelle 704 bis 766 gewählt, für die Wolke 832 bis 894.

Weil es auch andere Bereiche hätten sein können, muß dem Commodore 64 über deren Plazierung im Speicher eine Mitteilung gemacht werden. Dafür sind die Speicherstellen von 2040 bis 2047 verantwortlich. Sie geben den Ort des Sprite-Aufbaus im Speicher jeweils als Vielfache von 64 an. Die Zeile 510 mit POKE 2040,11 besagt also, daß das Sprite mit der Nummer 1 im Speicher ab der Speicherstelle 11∗64=704 abgelegt ist. Sprite 2 erhält auf diese Weise in Zeile 520 mit POKE 2041,13 die Wolkenstruktur, die im Speicher ab der Stelle 13∗64=832 steht.

Daß man verschiedenen Sprites den gleichen Speicherbereich zuordnen kann, zeigt Zeile 530: das Sprite 3 ist mit POKE 2042,11 ebenfalls ein Ballon. Beide Sprites lassen sich später natürlich trotzdem völlig unabhängig voneinander bewegen und verschieden einfärben.

Mit der Durchnumerierung ist gleichzeitig festgelegt, wie die Sprites bei einem Zusammentreffen abgebildet werden: Der erste Ballon (Sprite 1) fliegt vor der Wolke entlang und ist auch näher zum Betrachter als der zweite Ballon (Sprite 3). Die Wolke (Sprite 2) befindet sich zwischen den Ballons und wird vom ersten verdeckt, läßt den zweiten aber hinter sich vorbeiziehen.

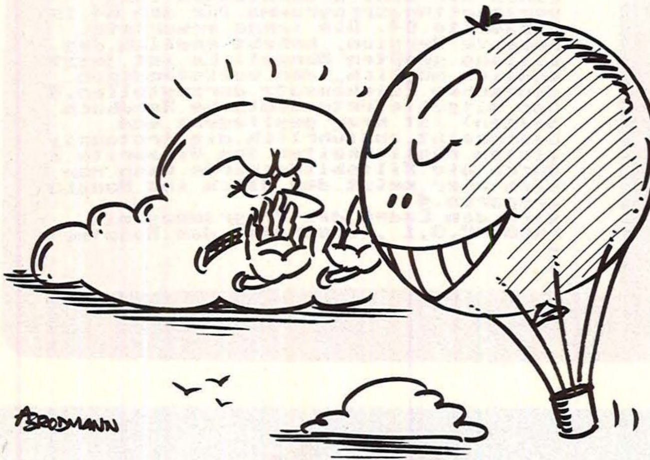
Sprites mit Vorrang

Nahezu perfekt wird die räumliche Illusion dadurch, daß Sprites mit niedrigerer Priorität durch Lücken im verdeckenden Sprite hindurch noch bruchstückhaft zu sehen sind. Zur Veränderung der Rangfolge können die Sprites innerhalb der Speicherstellen 2040 bis 2047 beliebig umgeordnet werden.

Für die Farben sind die Speicherstellen 53287 bis 53294 in der Reihenfolge der Sprite-Nummern zuständig. Dazu können die Zahlen 1 bis 16 entsprechend der 16 Farben des C 64 zugeordnet werden. In den Zeilen 610 bis 630 wurden Farben gewählt, die sich untereinander und vom Hintergrund gut abheben.

Da die Farben und auch die meisten anderen Eigenschaften der Sprites alle durch Speicherstellen nach 53248 festgelegt werden, wurde zur Vorbereitung in der Programmzeile 150 der Variablen V der Wert 53248 zugewiesen, damit diese Speicherstellen in Kurzform

Scan: S. Höltgen



(nur durch Addition kleiner Werte zu V) angesprochen werden können. Zum Beispiel die Speicherstellen 53271 und 53277: Sie sind für eine Streckung der Sprites in x- bzw. y-Richtung zuständig. Ein Sprite, das auf diese Weise in beide Richtungen gestreckt wurde, belegt danach eine viermal so große Fläche.

Peek und Poke

Jedes bit der beiden Speicherstellen ist für ein Sprite verantwortlich; bit 0 von 53271 beispielsweise entscheidet darüber, ob Sprite 1 in x-Richtung gestreckt ist, bit 1 von 53271 gibt an, ob Sprite 2 in x-Richtung gestreckt ist usw. Zu diesem Zweck müssen die jeweiligen bit gesetzt sein. In den Zeilen 710 und 720 des Listings von Tabelle 18 wurden alle drei Sprites in beiden Richtungen gestreckt, den entsprechenden Speicherstellen 53271 und 53277 mußte dazu der Wert 1+2+4=7 zugewiesen werden.

Auf die gleiche Weise ergibt sich der Wert 7, wenn man veranlassen will, daß alle drei Sprites sichtbar sein sollen, worüber die Speicherstelle 53269 entscheidet.

Die x- und y-Koordinaten der einzelnen Sprites auf dem Bildschirm sind ab 53248 gespeichert: 53248 enthält die x-Koordinate von Sprite 1, 53250 die x-Koordinate von Sprite 2 usw.

Im Listing ab Zeile 900 wurden solche Bewegungen gewählt, die am augenfälligsten demonstrieren können, welches Sprite von welchem verdeckt wird.

In Tabelle 17 sind die wichtigsten Adressen zur Sprite-Behandlung in einer Übersicht zusammengestellt. Die Adressen zur Abfrage einer Kollision zwischen Sprites und die Adressen zur Anzeige, daß sich ein Sprite im rechten Teil des Bildschirmes befindet, sind nicht aufgeführt.

In der nächsten Folge des BASIC-Kurses soll näher darauf eingegangen werden, wie im Vergleich zum C 64 die Sprite-Behandlung beim TI-99/4A und beim Atari ausfällt.

Commodore 64

```
100 REM *** VORBEREITUNGEN
110 PRINT CHR$(147)
150 LET V=53248
200 REM *** AUFBAU
210 FOR I=0 TO 62
220 READ B
230 POKE 704+I,B
240 NEXT I
250 DATA 0,127,0,1,255,192,3,255,224
260 DATA 3,231,224,7,219,240,7,223,240
270 DATA 7,219,240,3,231,224,3,255,224
280 DATA 3,225,224,2,255,160,1,127,64
290 DATA 1,62,64,0,156,128,0,156,128
300 DATA 0,73,0,0,73,0,0,62,0
310 DATA 0,62,0,0,62,0,0,28,0
320 FOR I=0 TO 62
330 READ B
340 POKE 832+I,B
350 NEXT I
360 DATA 0,0,0,0,1,240,0,7,252
370 DATA 0,127,254,3,255,254,15,
    255,255
380 DATA 31,255,255,63,255,255,127,
    255,255
390 DATA 127,255,255,248,255,255,
    224,127,128
400 DATA 224,28,0,224,24,0,112,48,0
410 DATA 127,240,0,63,224,0,15,192,0
420 DATA 3,128,0,0,0,0,0,0,0
500 REM *** DURCHNUMERIERUNG
510 POKE 2040,11
520 POKE 2041,13
530 POKE 2042,11
600 REM *** FARBEN
610 POKE V+39,8
620 POKE V+40,1
630 POKE V+41,13
700 REM *** VERGROESSERUNG
710 POKE V+23,7
720 POKE V+29,7
800 REM *** SICHTBARKEIT
810 POKE V+21,7
900 REM *** BEWEGUNG
910 FOR I=50 TO 150
920 POKE V,I*1.5
930 POKE V+1,(200-I)*1.3
940 POKE V+2, I*2.4-120
950 POKE V+3,85
960 POKE V+4,I
970 POKE V+5,I
980 NEXT I
990 GOTO 910
```

Tabelle 18: Listing zur Sprite-Demonstration

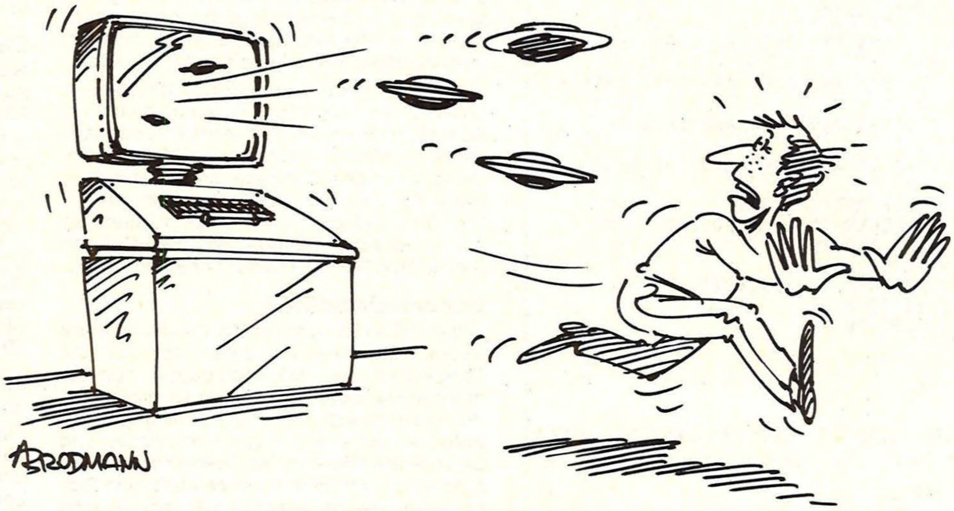
	Erste Stelle des Aufbaus geteilt durch 64	Farben	Streckung in x-Richtung	Streckung in y-Richtung	Sichtbarkeit	x-Koordinate auf dem Schirm	y-Koordinate auf dem Schirm
Sprite 1	2040	53287	bit 0 in 53271	bit 0 in 53277	bit 0 in 53269	53248	53249
Sprite 2	2041	53288	bit 1 in 53271	bit 1 in 53277	bit 1 in 53269	53250	53251
Sprite 3	2042	53289	bit 2 in 53271	bit 2 in 53277	bit 2 in 53269	53252	53253
Sprite 4	2043	53290	bit 3 in 53271	bit 3 in 53277	bit 3 in 53269	53254	53255
Sprite 5	2044	53291	bit 4 in 53271	bit 4 in 53277	bit 4 in 53269	53256	53257
Sprite 6	2045	53292	bit 5 in 53271	bit 5 in 53277	bit 5 in 53269	53258	53259
Sprite 7	2046	53293	bit 6 in 53271	bit 6 in 53277	bit 6 in 53269	53260	53261
Sprite 8	2047	53294	bit 7 in 53271	bit 7 in 53277	bit 7 in 53269	53262	53263

Tabelle 17: Übersicht der wichtigsten Speicherstellen für die Eigenschaften und die Steuerung der Sprites beim C 64

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore, Texas Instruments 99/4A und Atari



In der vorigen Folge des BASIC-Kurses wurde erklärt, wie man beim C64 mit Sprites umgeht. Es zeigte sich, daß Sprites große Figuren sind, für die der Rechner nützliche Hilfen zur Bewegung, gegenseitiger Verdeckung, Vergrößerung, Färbung und zur Kollisionsabfrage bietet.

Von den im BASIC-Kurs berücksichtigten Rechnern sind Sprites außer beim C64 noch beim TI-99/4A mit Extended-BASIC und beim Atari möglich. Für diese beiden Rechner sollen die wichtigsten Möglichkeiten des Sprite-Aufbaus und der Sprite-Beeinflussung im Vergleich zum C64 jetzt vorgestellt werden.

Sprites im Quadrat beim TI-99/4A

Um beim TI-99/4 A anzugeben, wie ein Sprite aussehen soll, benutzt man (wie bei der Definition eigener Zeichen) das Unterprogramm CHAR. Man muß sich dabei aber nicht auf ein einzelnes Zeichen beschränken, sondern kann insgesamt 32 Byte in hexadezimaler Form angeben. Damit hat man dann nicht nur das Zeichen mit der angegebenen Code-Zahl, sondern auch die nächsten drei belegt.

Auf diese Weise werden in der Programmzeile 210 der Tabelle 19 die Zeichen mit den Code-Zahlen 96 bis 99 alle zu gefüllten Quadraten. Wenn die vier Zeichen zusammen als ein Sprite angesprochen werden, dann hat das Sprite die Form eines großen Quadrates, das sich folgendermaßen aus den vier Zeichen zusammensetzt:
erstes Zeichen links oben
zweites Zeichen links unten
drittes Zeichen rechts oben
viertes Zeichen rechts unten

Die Programmzeile 220 baut einen großen Rahmen mit einem quadratischen Loch in der Mitte in den Zeichen mit den Code-Zahlen 100 bis 103 auf.

In den Zeilen 510 bis 530 werden mit dem Unterprogramm SPRITE die beiden aufgebauten Muster als Sprites angesprochen. Dort erhält jedes Sprite hinter einem Doppelkreuz zunächst eine Nummer (da insgesamt 28 Sprites verwaltet werden können, sind Zahlen bis 28 möglich).

Die nächste Zahl gibt an, ab welcher Code-Zahl Zeichen für das Sprite benutzt werden sollen. Im Programm sind die Sprites 1 und 3 große gefüllte Quadrate; Sprite 2 ist ein Rahmen mit quadratischem Loch in der Mitte.

Die nächste Angabe legt mit einer Zahl zwischen 1 und 16 die Farbe des Sprites fest. Die Anfangsposition des Sprites wird mit den Angaben 4 und 5 vorgegeben; zunächst wird die Reihenzahl zwischen 1 und 192, dann die Spaltenzahl zwischen 1 und 256 aufgeführt.

Etwas größer bitte!

Erst das Unterprogramm MAGNIFY bestimmt, ob für die Sprites vier Zeichen oder nur ein Zeichen verwendet werden. Mit den Zahlen 1 und 2 findet für jedes Sprite jeweils nur ein Zeichen Verwendung, mit den Zahlen 3 und 4 jeweils vier. Die Zahlen 2 und 4 vergrößern alle Sprites außerdem noch sowohl in x- als auch in y-Richtung auf das Doppelte. Mit der Zahl 4 wird also bei jedem Sprite auf vier Zeichen zugegriffen und jedes Sprite vergrößert dargestellt.

In den Programmzeilen 920 bis 960 ist das Unterprogramm MOTION für die Bewegung der Sprites verantwortlich. Nach der Nummer des Sprites hinter einem Doppelkreuz folgt die Geschwindigkeit in y-Richtung, dann die in x-Richtung. Die Zahlen müssen zwischen −128 und 127 liegen; je weiter die Werte von Null entfernt sind, um so schneller bewegt sich das jeweilige Sprite.

Es ist eigentlich nicht erforderlich, das Unterprogramm MOTION dreimal aufzurufen; man kann die Angaben für alle Sprites auch in einem MOTION-Aufruf aneinanderreihen. Das gleiche gilt auch für das SPRITE-Unterprogramm. Es ist auch möglich, die Geschwindigkeit schon im SPRITE-Aufruf hinter der Startposition anzugeben.

Scan: S. Höltgen

In Streifen beim Atari

Beim Atari haben Sprites einen anderen Namen: sie heißen dort Players. Ihre Besonderheit ist, daß sie nur 8 bit breit sind, dafür aber die ganze Höhe des Bildschirmes ausnutzen, also 256 bit hoch sein können.

Nach der Wahl der Grafikbetriebsart in der Programmzeile 110 muß durch Änderung der Speicherstelle 106 zunächst Platz für den Player-Aufbau geschaffen werden. Der Wert 62 für die Speicherstelle 559 legt für die Sprites eine doppelte Auflösung fest.

In der Programmzeile 150 wird der Variablen V der Wert 53248 zugewiesen, damit in den folgenden POKE-Anweisungen Adressen nach 53248 in Kurzform angesprochen werden können.

In der Programmzeile 160 ist die Zahl 3 an der Speicherstelle 53277 dafür verantwortlich, daß sowohl Players als auch Missiles (die kleinen Brüder der Players mit einer Breite von 2 bit) dargestellt werden können.

Die Anfangadresse der Players im Speicher muß an der Speicherstelle 54279 angegeben werden; dies geschieht in Programmzeile 170. Der Beginn des Player-Aufbaus darf bei doppelter Auflösung in Wirklichkeit jedoch erst 1024 Byte danach erfolgen, was in Zeile 210 berücksichtigt wird.

Vertikale Bewegung nicht vorgesehen

Die Variable Y gibt über die Lage des Objektes innerhalb der Player-Säulen Auskunft; im abgedruckten Programm soll die gesamte Säulenhöhe ausgenutzt werden; deshalb erhält Y den Wert Null.

Die Zeilen 230 bis 270 bauen drei Players auf: zwei lange Streifen mit und einen ohne Lücke. Insgesamt können beim Atari vier Players gleichzeitig verwaltet werden.

Für die Farbe der Players sind die Speicherstellen 704 bis 707 verantwortlich, die Zahlen zwischen 0 bis 255 aufnehmen können.

Die Bewegung wird für jeden Player durch jeweils nur eine Speicherstelle gesteuert, welche die horizontale Position (also die Position in x-Richtung) festlegt; dies sind die Speicherstellen 53248 bis 53251. Für vertikale Bewegungen (in y-Richtung) bietet der Atari keine eigene Speicherstelle an.

Vertikale Bewegungen sind aber dadurch möglich, daß man den Player selbst ändert: das Objekt wird innerhalb der Player-Säule nach oben oder unten verrückt. In BASIC ist eine solche Operation sehr zeitaufwendig; deshalb benutzt man dazu meist eine Maschinen-Code-Routine.

Die Prioritäten zwischen den einzelnen Playern sind durch die Reihenfolge im Speicher vorgegeben. Die Programmbeispiele wurden jeweils so gewählt, daß deutlich wird, welches Sprite (beziehungsweise welcher Player) durch welchen anderen verdeckt werden kann.

Der BASIC-Kurs wird im nächsten Heft fortgesetzt.

Commodore 64	TI-99/4A mit Extended-BASIC	Atari
100 REM *** VORBEREITUNGEN	100 REM *** VORBEREITUNGEN	100 REM *** VORBEREITUNGEN
110 PRINT CHR$(147)	110 CALL CLEAR	110 GRAPHICS 3+16
		120 LET A=PEEK(106)−16
		130 POKE 106,A
		140 POKE 559,62
150 LET V=53248		150 LET V=53248
		160 POKE V+29,3
		170 POKE V+1031,A
200 REM *** AUFBAU	200 REM *** AUFBAU	200 REM *** AUFBAU
210 FOR I=0 TO 62	210 CALL CHAR (96,"FFFFFF FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF")	210 LET BEG=(A+4)*256
220 READ B	220 CALL CHAR(100,"FFFFFFFF0 F00F0F0F0F0F 0F0FFFFFF FFFFFFFFFF0F0 F0F0F0F0F0F0FF FFFFFF")	220 LET Y=0
230 POKE 704+I,B		230 FOR I=0 TO 2
240 NEXT I		240 FOR J=0 TO 255
250 DATA 0,127,0,1,255,192,3,255,224		250 POKE BEG+I*256+Y+J,199+ ABS(I−1)*56
260 DATA 3,231,224,7,219,240,7,223,240		260 NEXT J
270 DATA 7,219,240,3,231,224, 3,255,224		270 NEXT I
280 DATA 3,225,224,2,255,160,1,127,64		
290 DATA 1,62,64,0,156,128,0,156,128		
300 DATA 0,73,0,0,73,0,0,62,0		
310 DATA 0,62,0,0,62,0,0,28,0		
320 FOR I=0 TO 62		
330 READ B		
340 POKE 832+I,B		
350 NEXT I		
360 DATA 0,0,0,0,1,240,0,7,252		
370 DATA 0,127,254,3,255,254,15, 255,255		
380 DATA 31,255,255,63,255,255,127, 255,255		
390 DATA 127,255,255,248,255,255, 224,127,128		
400 DATA 224,28,0,224,24,0,112,48,0		
410 DATA 127,240,0,63,224,0,15,192,0		
420 DATA 3,128,0,0,0,0,0,0,0		
500 REM *** DURCHNUMERIERUNG	500 *** REM NUMERIERUNG,FARBE, ANFANGSPOSITION	
510 POKE 2040,11	510 CALL SPRITE (#1,96,4,150,50)	
520 POKE 2041,13	520 CALL SPRITE (#2,100,7,85,1)	
530 POKE 2042,11	530 CALL SPRITE (#3,96,5,50,50)	
600 REM *** FARBEN		600 REM *** FARBEN
610 POKE V+39,8		610 POKE 704,55
620 POKE V+40,1		620 POKE 705,111
630 POKE V+41,13		630 POKE 706,137
700 REM *** VERGROESSERUNG	700 REM *** VERGROESSERUNG	
710 POKE V+23,7	710 CALL MAGNIFY(4)	
720 POKE V+29,7		
800 REM *** SICHTBARKEIT		
810 POKE V+21,7		
900 REM *** BEWEGUNG	900 REM *** BEWEGUNG	900 REM *** BEWEGUNG
910 FOR I=50 TO 150		910 FOR I=0 TO 228
920 POKE V,I*1.5	920 CALL MOTION(#1,−13,15)	920 POKE V,228−1
930 POKE V+1,(200−I)*1.3		
940 POKE V+2, I*2.4−120	940 CALL MOTION(#2,0,24)	940 POKE V+1,150
950 POKE V+3,85		
960 POKE V+4,I	960 CALL MOTION(#3,10,10)	960 POKE V+2,I
970 POKE V+5,I		
		975 FOR J=1 TO 30:NEXT J
980 NEXT I		980 NEXT I
990 GOTO 910	990 GOTO 990	990 GOTO 910

Tab. 19: Listings zur Sprite-Demonstration

Scan: S. Höltgen

chen BASIC-Programm, welches die Tasteneingaben an die Schnittstelle schickt und die Zeichen von der Schnittstelle auf den Bildschirm „printet". Weil Commodore einen etwas eigensinnigen „ASCII-Zeichensatz" verwendet, sich also nicht an das Übliche hält, sind ein paar Korrekturen nötig, damit die Groß- und Kleinschreibung und das Löschen (DELete) von Zeichen funktionieren (Zeilen 8, 9, 10, 12–15).

Zum Testen

Dieses Programm erfüllt fürs erste seinen Zweck. Wer mehr will, als nur einzelne Zeichen von der Tastatur abzuschicken, wer zum Beispiel größere Texte oder Programme übertragen will, kann auf diesem prinzipiellen Programm aufbauen. Öffnet man die Schnittstelle mit den richtigen Übertragungsparametern von Hand (wie in Zeile 5) und gibt CMD1, so kann ohne weiteres auch ein Programmlisting übertragen werden. Allerdings ist es vorher ratsam, die Commodore-Sonderzeichen wie CLR/HOME durch ihre CHR$-Äquivalente (CHR$(147) zu ersetzen, sonst gehen bei der Übertragung Zeichen verloren.

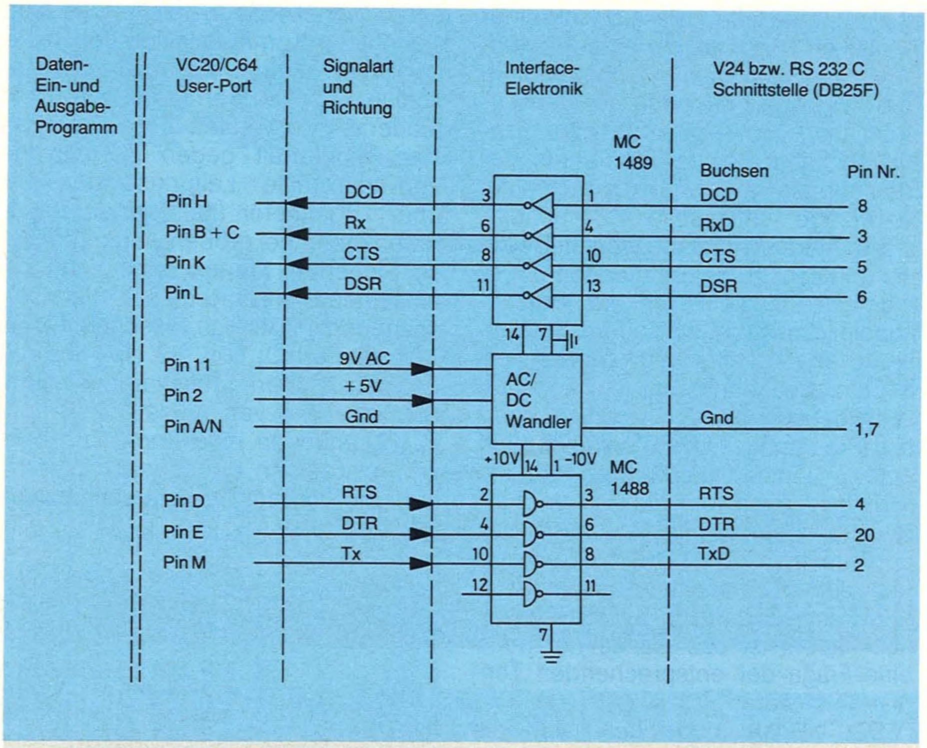
Im übrigen gibt es mittlerweile im Handel sehr leistungsfähige Terminalprogramme. Diese gestatten es, Texte vorzuschreiben und zu bearbeiten, abzuspeichern und als Text-File über die Schnittstelle abzuschicken. Ebenso können damit Texte oder Programme, die empfangen werden, in den Speicher geschrieben und nachher auf Kassette oder Diskette „abesaved" oder gedruckt werden.

Für Bastler

Wer noch kein Modem besitzt, sich aber schon durch das Basteln einer Schnittstelle darauf vorbereiten möchte, braucht zum Test dieser Schnittstelle noch einen heißen Tip: Mit einem Nullmodem kann man die fertig aufgebaute V.24-Schnittstelle testen. Ein Nullmodem besteht aus einem DB-25-Stecker (oder Buchse), bei dem die Pins 2 + 3, 4 + 5 und 6 + 20 mit je einer Drahtbrücke kurzgeschlossen sind. Der Rechner „gibt sich so selbst die Hand", und die getippten Zeichen erscheinen auch auf dem Bildschirm.

Wer ein Modem besitzt, kann sich sein Interface und das Verbindungskabel in einer Billig- oder in einer Luxusversion aufbauen. Ein Flachbandkabel mit Anpreßstecker kommt billiger als eine DB-25-Buchse für Leiterplattenmontage, die man ins Interface einbaut. Im zweiten Fall erhält man eine normale V.24-Schnittstellenbuchse an der Rechnerrückseite, wie bei jedem „gescheiten" größeren Computer. Das separate Verbindungskabel zu einem kommerziellen Akustikkoppler braucht dann parallel durchgehende und nicht überkreuzte Adern.

Hans-Peter Kroll



Schaltplan des Interface zur Pegelanpassung TTL-seriell auf V.24-Norm für den VC 20 und C 64.

Zur Erzeugung einer positiven und einer negativen Spannung von ca. plus/minus 10 Volt verwendet man 3 Dioden und 3 Elkos mit 100 F in Einweg-Gleichrichter-Schaltung. Die ICs sind speziell für solche Zwecke vorgesehene V.24-Leitungsempfänger und Leitungstreiber. Sie sind kurzschlußfest und garantieren die durch die Norm geforderten Werte.

Um den Commodore Software-Fehler provisorisch zu beheben, legt man den Pin D des User-Ports oder den Pin 2 des MC 1488 auf Masse.

Stückliste für das V.24-Interface

Lochrasterplatine 32 × 62 mm (2,54-mm-Raster)
24-poliger User-Port-Stecker und 2 Kodierkeile
DB-25-Stecker mit Flachbandkabel oder DB-25-Buchse für Printplattenmontage (2,77-mm-Raster)
Alu-Gehäuse 72 × 36 × 18 mm (verkleinert)
2 Abstandsbolzen M3 × 5
2 Blechschrauben ca. 2,5 × 12,2
Schrauben M3 × 10 und 6 Muttern M3
ca. 1 m isolierter Schaltdraht, Cu 0,5 mm Durchmesser
3 ELKOs 100 µ F/16 V
3 Dioden 1N4001
1 MC 1488 (SN 75188N)
1 MC 1489 (SN 75189AN)
Preis der Bauteile ca. 30 Mark
Hilfsmittel:
Lötkolben max. 30 Watt, Lot, Laubsäge, 3 mm Bohrer und Handwerkszeug

Scan: S. Höltgen

Programmieren in BASIC

In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair-Spectrum, Commodore, Texas Instruments 99/4A und Atari

Bei den musikalischen Fähigkeiten bestehen große Unterschiede zwischen den einzelnen Home-Computer. Auch die Art und Weise, wie man diese Fähigkeiten entfaltet, weicht stark voneinander ab.

Während Spectrum, TI-99/4A und Atari die Tonerzeugung durch BASIC-Befehle erleichtern, sind bei den Commodore-Rechnern bestimmte Speicherstellen (Register) für die Musik zuständig. Der VC 20 und der C 64 machen es also erforderlich, daß man sich die Nummern dieser Speicherstellen entweder gut einprägt oder jedes Mal aus einer Tabelle heraussucht.

C-Dur-Tonleiter

Die abgedruckte Tabelle für den C 64 ist sehr umfangreich, weil er zusammen mit drei Tongeneratoren vielfältige Möglichkeiten zur Klanggestaltung bietet, die schon eine Art Synthesizer darstellen.

Der Spectrum kann nur einstimmige Lieder erklingen lassen, weil er nur einen Tongenerator besitzt. Die wichtigsten Unterschiede zwischen Tönen bestehen in ihrer Höhe und ihrer Dauer. Das sind auch die einzigen Werte, die beim Spectrum in dieser Reihenfolge hinter BEEP angegeben werden müssen.

Da als kleinstmöglicher Tonhöhenunterschied ein Halbtonschrift üblich ist, werden beim Spectrum diese Halbtöne aneinandergereiht und durchnumeriert.

Der als „mittleres C" bezeichnete Ton hat beim Spectrum den Tonhöhewert Ø. Berücksichtigt man nun, daß eine Durtonleiter bis auf zwei Ausnahmen aus sieben Ganztonschritten besteht (der dritte und der siebte sind Halbtonschritte), so ergibt sich zum Beispiel für die C-Dur-Tonleiter über dem mittleren C die Zahlenreihe 0,2,4,5,7,9,11,12.

Man ist beim Spectrum aber nicht an diese ganzen Zahlen gebunden, sondern kann beliebige Dezimalzahlen angeben. Dies kommt bei zwei recht gegensätzlichen Anwendungen in Frage: dem absichtlichen Verstimmen des Home-Computers oder der Benutzung der natürlichen Tonleiter.

Zwischentöne

Bei allen anderen hier aufgeführten Home-Computern sind als Tonhöhe nur ganze Zahlen möglich, oder es wird bei Angabe von Dezimalzahlen vom Rechner zu ganzen Zahlen hin gerundet. Jedoch sind die Werte der Halbtöne so locker gefügt, daß auch noch genug Platz für Zwischentöne bleibt.

Der einzige der vier Home-Computer, der sich an dem physikalischen Begriff Frequenz zur Beschreibung einer Tonhöhe orientiert und die Angabe in der Einheit Hertz erwartet, ist der TI-99/4A. Dort dient zur Tonerzeugung ein Unterprogramm, das mit CALL SOUND aufgerufen wird.

VC 20	Stimme 1	Stimme 2	Stimme 3
Tonhöhe	36874	36875	36876
Lautstärke		36878	

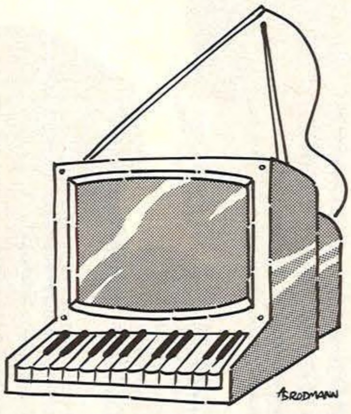
Commodore 64	Stimme 1	Stimme 2	Stimme 3
Tonhöhe	54272 54273	54279 54280	54286 54287
Oberton-Zusammensetzung	54274 54275	54281 54282	54288 54289
Kontroll-bits	bit 0 bis 3 von 54276	bit 0 bis 3 von 54283	bit 0 bis 3 von 54290
Wellenform	bit 4 bis 7 von 54276	bit 4 bis 7 von 54283	bit 4 bis 7 von 54290
Abklingzeit	bit 0 bis 3 von 54277	bit 0 bis 3 von 54284	bit 0 bis 3 von 54291
Einschwingzeit	bit 4 bis 7 von 54277	bit 4 bis 7 von 54284	bit 4 bis 7 von 54291
Haltepegel	bit 0 bis 3 von 54278	bit 0 bis 3 von 54285	bit 0 bis 3 von 54292
Ausklingzeit	bit 4 bis 7 von 54278	bit 4 bis 7 von 54285	bit 4 bis 7 von 54292
Lautstärke		54296	

Scan: S. Höltgen

Als Parameter werden dem Unterprogramm zunächst die Tondauer in Millisekunden, dann die Tonfrequenz und schließlich die Lautstärke übergeben; es können dann Frequenz- und Lautstärkeangaben für die anderen beiden Tongeneratoren folgen.

SOUND heißt auch der BASIC-Befehl, der dem Atari Töne entlockt. Er benötigt zunächst die Nummer des Generators (0 bis 3), dann die Tonhöhe, eine Zahl für die Reinheit des Tones (0 bis 15) und die Lautstärke (0 bis 15).

Bei der Tonreinheit des Atari ist zu beachten, daß ungerade Zahlen den betroffenen Generator abschalten. Die Tonhöhe wird im Gegensatz zu den anderen Home-Computern mit steigenden Werten niedriger. Ein Ton erklingt beim Atari solange, bis er mit einem anderen SOUND-Befehl wieder abgeschaltet wird.



Ebenfalls keine Vorgabe der Tondauer sehen die beiden Commodore-Rechner vor. Die Tonhöhen der drei Stimmen wird beim VC 20 durch die Register in den Speicherstellen 36874 bis 36876 bestimmt. Speicherstelle 36878 gibt deren gemeinsame Lautstärke an.

Beim C 64 ist für die Tonhöhe jeweils ein Doppelregister zuständig, wobei das niederwertige Byte (an der Speicherstelle mit der niedrigeren Nummer) die Feinabstimmung und das höherwertige Byte (an der Speicherstelle mit der höheren Nummer) die Grobabstimmung übernehmen. Die Informationen für die drei Tongeneratoren stehen dort in den Speicherstellen 54272 bis 54292; die gemeinsame Lautstärke in 54296.

Probieren geht über Studieren

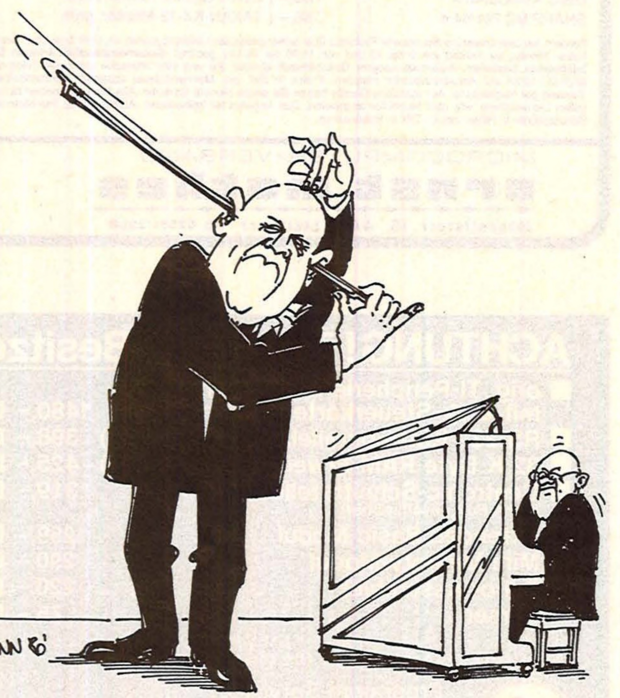
Die entsprechenden Register müssen beim Commodore jeweils mit einem POKE-Befehl eingestellt werden. Für alle Home-Computer gilt: man sollte sich viel Zeit zum Experimentieren nehmen und durch Abändern der Register-Inhalte auf Entdeckungsreise gehen. Wichtiges Utensil beim C 64, das leicht vergessen wird: bit 0 des Kontroll-bit-Wellenform-Registers, es muß gesetzt sein, damit die entsprechende Stimme erklingt, die in diesen Registern stehenden Zahlen müssen also ungerade sein.

Zum Aufbruch können Sie das abgedruckte Programm „Gehör-Schulung" benutzen. Dort wird in den Zeilen 10 bis 110 zunächst die C-Dur-Tonart mit einem Tonumfang von drei Oktaven im Feld V gespeichert. Feld N$ nimmt danach die Bezeichnungen von acht Intervallen (Tonschritten) auf.

In den Zeilen 220 bis 240 werden zufällig zwei Töne ausgewählt, die maximal eine Oktave auseinanderliegen. Ab Zeile 300 wird der Benutzer aufgefordert, das Intervall zu bestimmen; dabei stehen die Zahlen 0 bis 7 für Prime bis Oktave. Es folgt eine Mitteilung über die Richtigkeit der Eingabe und ein neues Intervall wird zur Analyse vorgestellt. *Der BASIC-Kurs wird im nächsten Heft fortgesetzt.*

Spectrum	VC 20	C 64	TI-99/4A	Atari
10 DIM U(22)	10 DIM U(22)	10 DIM U(22,2)	10 DIM U(22)	10 DIM U(22)
20 FOR J=1 TO 22	20 FOR J=1 TO 22	20 FOR J=1 TO 22	20 FOR J=1 TO 22	20 FOR J=1 TO 22
30 READ U(J)	30 READ U(J)	30 READ U(J,1)	30 READ U(J)	30 READ X: U(J)=X
		40 READ U(J,2)		
50 NEXT J	50 NEXT J	50 NEXT J	50 NEXT J	50 NEXT J
60 DATA −12,−10,−8,−7	60 DATA 135,147,159,163	60 DATA 4,90,4,226,5,123,5,207	60 DATA 131,147,165,175	60 DATA 243,217,193,182
70 DATA −5,−3,−1,−0	70 DATA 175,183,191,195	70 DATA 6,133,7,81,8,55,8,180	70 DATA 196,220,247,262	70 DATA 162,144,128,121
80 DATA 2,4,5,7	80 DATA 201,207,209,215	80 DATA 9,196,10,247,11, 158,13,10	80 DATA 294,330,349,392	80 DATA 108,96,91,81
90 DATA 9,11,12,14	90 DATA 219, 223, 225,228	90 DATA 14,162,16,109,17, 103,19,137	90 DATA 440,494,523,587	90 DATA 72,64,60,53
100 DATA 16,17,19,21	100 DATA 231,232,235,237	100 DATA 21,237,23,59,26, 20,29,69	100 DATA 659,698,784,880	100 DATA 47,45,40,35
110 DATA 23,24	110 DATA 239,240	110 DATA 32,219,34,207	110 DATA 988,1047	110 DATA 31,29
120 DIM N$ (8,7)	120 DIM N$(8)	120 DIM N$(8)		120 DIM N$(56): DIM X$(7)
130 FOR J=1 TO 8	130 FOR J=1 TO 8	130 FOR J=1 TO 8	130 FOR J=1 TO 8	130 FOR J=1 TO 8
140 READ N$ (J)	140 READ N$(J)	140 READ N$(J)	140 READ N$(J)	140 READ X$:N$ (7 * J−6,7 * J)=X$
150 NEXT J	150 NEXT J	150 NEXT J	150 NEXT J	150 NEXT J
160 DATA "PRIME", "SEKUNDE","TERZ", "QUARTE"	160 DATA "PRIME", "SEKUNDE","TERZ", "QUARTE"	160 DATA "PRIME", "SEKUNDE","TERZ", "QUARTE"	160 DATA "PRIME", "SEKUNDE","TERZ", "QUARTE"	160 DATA PRIME−−, SEKUNDE, TERZ−−−, QUARTE−
170 DATA "QUINTE", "SEXTE","SEPTIME", "OKTAVE"	170 DATA "QUINTE", "SEXTE","SEPTIME", "OKTAVE"	170 DATA "QUINTE", "SEXTE","SEPTIME", "OKTAVE"	170 DATA "QUINTE", "SEXTE","SEPTIME", "OKTAVE"	170 DATA QUINTE−, SEXTE−−, SEPTIME, OKTAVE−
	180 SI=36874	180 SI=54272		
		190 POKE SI+4,17		
		200 POKE SI+11,17		
	210 POKE SI+4,10	210 POKE SI+24,15		
220 LET T=8+INT (RND * 8)	220 T=8+INT(RND(1) * 8)	220 T=8+INT(RND(1) * 8)	220 T=8+INT(RND * 8)	220 T=8+INT(RND(1) * 8)
230 LET I=INT (RND * 8)	230 I=INT(RND(1) * 8)	230 I=INT(RND(1) * 8)	230 I=INT(RND * 8)	230 I=INT(RND(1) * 8)+1
240 LET R=INT (RND * 2) * 2−1	240 R=INT(RND(1) * 2) * 2−1	240 R=INT(RND(1) * 2) * 2−1	240 R=INT(RND * 2) * 2−1	240 R=INT(RND(1) * 2) * 2−1
250 BEEP 1, U (T)	250 POKE SI,U(T)	250 POKE SI,U(T,2)	250 CALL SOUND (1000,U(T), 20, U(T+R * I),20)	250 SOUND 0,U(T),14,10
		260 POKE SI+1,U(T,1)		
270 BEEP 1, U (T+R * I)	270 POKE SI+1,U(T+R * I)	270 POKE SI+7,U(T+R * I,2)		270 SOUND 1, U(T+R * (I−1)),14,10
		280 POKE SI+8,U(T+R * I,1)		
		290 POKE SI+24,0		
300 PRINT "WELCHES INTERVALL?"	300 PRINT "WELCHES INTERVALL?"	300 PRINT "WELCHES INTERVALL?"	300 PRINT "WELCHES INTERVALL?"	300 PRINT "WELCHES INTERVALL?"
310 INPUT F	310 INPUT F	310 INPUT F	310 INPUT F	310 INPUT F
320 IF F<0 OR F>7 THEN GO TO 310	320 IF F<0 OR F>7 THEN 310	320 IF F<0 OR F>7 THEN 310	320 IF F<0 THEN 310	320 IF F<0 OR F>7 THEN 310
			330 IF F>7 THEN 310	
340 PRINT "DU MEINST: EINE ";N$(F+1)	340 PRINT "DU MEINST: "; N$(F+1)	340 PRINT "DU MEINST: EINE ";N$(F+1)	340 PRINT "DU MEINST: EINE ";N$(F+1)	340 PRINT "DU MEINST: EINE "; N$(F * 7+1, F * 7+7)
350 PRINT "RICHTIG IST: EINE ";N$(I+1)	350 PRINT "RICHTIG IST: "; N$(I+1)	350 PRINT "RICHTIG IST: EINE ";N$(I+1)	350 PRINT „RICHTIG IST: EINE ";N$(I+1)	350 PRINT "RICHTIG IST: EINE ";N$(I * 7−6, I * 7)
360 PRINT	360 PRINT	360 PRINT	360 PRINT	360 PRINT
370 GOTO 220	370 GOTO 220	370 GOTO 220	370 GOTO220	370 GOTO 220

Programmieren in BASIC



In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair Spectrum, Commodore, Texas Instruments 99/4A und Atari

Musik und Geräusche liegen nicht weit auseinander, und die Übergänge sind fließend. Auch bei Home-Computern befinden sich die für den Krach verantwortlichen Speicherstellen Tür an Tür mit denen für die Töne.

Beim VC 20 gibt es zusätzlich zum Stimmregister ein Geräuschregister. Es ist mit der Adresse 36877 versehen, während die Töne von 36874 bis 36876 beheimatet sind. Zuerst muß natürlich im Lautstärkeregister bei der Adresse 36878 die gewünschte Lautstärke eingestellt werden.

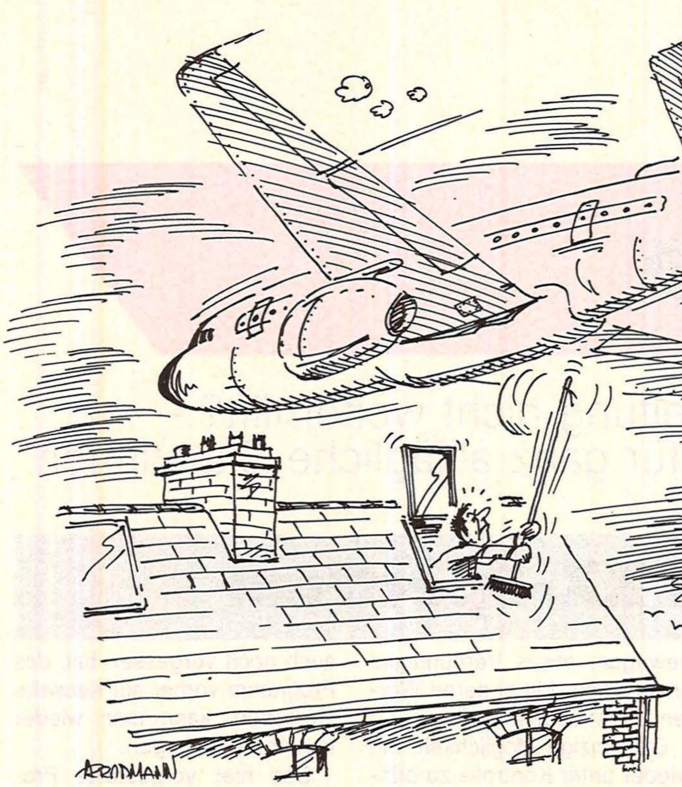
In den abgedruckten Listings soll für den VC 20 und drei weitere Rechner (der Spectrum sieht keine Geräusche vor) ein erster grober Überblick gegeben werden, welche Grundelemente zur Geräuscherzeugung jeweils vorrätig sind.

Die einzelnen Geräusche werden in diesen Programmen (soweit möglich) zusätzlich in verschiedenen Tonhöhen vorgestellt. Beim Programm für

VC 20	C 64	TI-99/4A	Atari
	10 POKE 54272,0		
	20 POKE 54296,15		
	30 FOR H=5 TO 20 STEP 5		
	40 FOR A=2 TO 12 STEP 5		
	50 FOR D=2 TO 12 STEP 5		
	60 FOR S=2 TO 12 STEP 5		
	70 FOR R=2 TO 12 STEP 5		
	80 PRINT"H=";H;" A=";A;		
	90 PRINT" D=";D;" S=";		
10 POKE 36878,6	100 PRINT S;" R=";R	10 FOR H=-8 TO -4	10 FOR H=40 TO 200 STEP 20
20 FOR H=128 TO 255	110 POKE 54276,129	20 CALL SOUND(2000,H,3)	20 FOR J=0 TO 8 STEP 2
30 POKE 36877,H	120 POKE 54273,H	30 NEXT H	30 SOUND 0,H,J,12
40 FOR W=1 TO 200:NEXT W	130 POKE 54277,16*A+D		40 FOR W=1 TO 200:NEXT W
50 NEXT H	140 POKE 54278,16*S+R		50 NEXT J
60 POKE 36878,0	150 FOR W=1 TO 3000:NEXT W		60 NEXT H
	160 POKE 54276,128		
	170 FOR W=1 TO 2000:NEXT W		
	180 NEXT R		
	190 NEXT S		
	200 NEXT D		
	210 NEXT A		
	220 NEXT H		

Tab. 22: Listings „Geräuschparade"

Scan: S. Höltgen



den C 64 werden außerdem die ADSR-Werte für den Klang variiert. Die Bezeichnung ADSR setzt sich aus den Anfangsbuchstaben der englischen Wörter für Einschwingzeit, Abklingzeit, Haltepegel und Ausklingzeit zusammen: Attack, Decay, Sustain und Release. Die Schleifenindizes im Programm sind mit diesen Anfangsbuchstaben bezeichnet worden.

Zunächst wird das niederwertige Byte der Tonhöhe, das in der Speicherstelle 54272 steht, auf Null gesetzt. Dieser Wert wird nicht verändert.

Das höherwertige Byte der Tonhöhe erhält bei der Vorstellung der Klänge jeweils vier verschiedene Werte. Alle Klang-Parameter werden, bevor das Geräusch zu hören ist, auf dem Bildschirm angezeigt, so daß der Zusammenhang zwischen Parameter-Variation und dem Geräusch deutlich wird. Als Wellenform wurde in allen Fällen „Rauschen" gewählt, was durch POKE 54276,129 erreicht wird.

Rauschen im Geräusche-Wald

Beim TI-99/4A entstehen Geräusche, wenn man negative Werte von −1 bis −8 für die Tonhöhe einsetzt. Zusätzliche Beeinflussung durch den dritten Tongenerator ist mit den Werten −4 und −8 möglich.

Beim Atari können durch Angabe von geraden Zahlen zwischen 0 und 8 an dritter Stelle hinter dem BASIC-Befehl SOUND mehr oder weniger starke Verzerrungen gewählt werden.

Bei allen Home-Computern darf nicht vergessen werden, daß im Zusammenspiel mit den Tongeneratoren (und beim C 64 mit Filtern, Pässen, Ring-Modulation und Synchronisation) weitere Klangeffekte erzielt werden können.

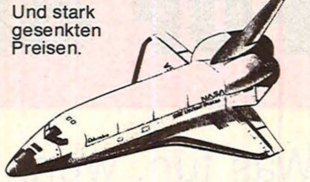
Der BASIC-Kurs wird im nächsten Heft fortgesetzt.

SPACE SHUTTLE

Warum nur im Fernsehen zuschauen! Holen Sie selbst in einer Simulation von 5 Flugphasen einen fehlerhaften Satelliten auf die Erde zurück. So erfolgreich, daß SPACE SHUTTLE inzwischen auf 8 Homecomputern läuft.

Mit deutscher Anleitung.

Und stark gesenkten Preisen.



- ATARI 16K ● BBC/B
- CBM 64 ● DRAGON 32K
- ELECTRON ● 48K ORIC 1
- TANDY COLOUR 32K
- SPECTRUM 48K. Je 27,– DM.

Für den DRAGON und TANDY ist ein Joystick erforderlich. Für CBM 64 und ATARI bieten wir auch eine DISK-Version an. (Preis 37,– DM). Bei Bestellungen bitte Scheck mitschicken + 3,– DM Versandkosten.

Großzügige Händlerkonditionen. Mit Rücknahmegarantie.

Wenn Sie an Informationen über unser weiteres Angebot für CBM 64, DRAGON 32, SPECTRUM 16/48K interessiert sind, geben Sie Ihren Computertyp an und senden bitte 2,40 DM in Briefmarken ein.

MST
Micro Software Trading GmbH
KRAIENKAMP 7
2000 TANGSTEDT
TELEFON 0 41 09/96 17

MCPS

Computersysteme für Büro und Hobby, Software

Auszug aus unserem umfangreichen Lieferprogramm:

SHARP MZ 721 776,– **MZ 731 1099,–**

SHARP 3"-Floppy für MZ 721/731	579,–
80-Zeichenkarte mit CP/M für MZ 721/731	945,–
PC 1401 Pocketcomputer	228,–
CE 126 Kassetteninterface u. Drucker	179,–
PC 1260 Pocketcomputer	314,–

APPLE-II-Zubehör und kompatible Geräte

ASTRA II, 48 KB, alle IC gesockelt	1049,–
Floppylaufwerk Slimline anschlußfertig	648,–
Monitor Sanyo, 16 MHz, 12", orange/grün 305/	284,–
Monitor Ciaegi, 20 MHz, 12", div. Modelle ab	328,–
Video + RGB-Monitor Sanyo CD 3185, 14"	869,–

IBS-Interface und andere für APPLE (und Applebus)

16-K-RAM-Karte (Languagekarte)	139,–
Farbkarte PAL-Video oder RGB	169,–
64-KB-RAM-Karte m. Pseudodisk	399,–
256-KB-RAM-Karte m. Pseudodisk (superschnelles RAM-Floppy) 64–256 KB	844,– bis 1398,–
80-Zeichen-Karte mit Softschalter	268,–
80-Zeichen-Karte mit 64-KB-RAM für IIe	355,–
Z80-Karte ohne Software	165,–

EPSON-Drucker RX 80 mit Traktorführung 1119,–

RX 80 F/T m. Einzelblatteinzug u. Traktor	1298,–
FX 80 m. Einzelblatteinzug u. Traktor	1699,–
BX80 Schönschreibdrucker m. Grafik, Einzelbl.	998,–
Seikosha GP500A mit Interf. f. Spectrum	866,–
Seikosha GP500A m. Interf. SHARP MZ700+MZ80A	866,–
Seikosha GP100VC-Drucker für VC20/C64	598,–
Commodore C64	698,–
Commodore Floppy VC 1541	748,–
Sinclair Spectrum 48/16K	a.A.
Floppy-Disk für Spectrum, 386K	1298,–
Sinclair-16-K-RAM-Erweiterung für ZX 81	89,–

DISKY-Disketten, 1a-Qualität, doppelte Bitdichte

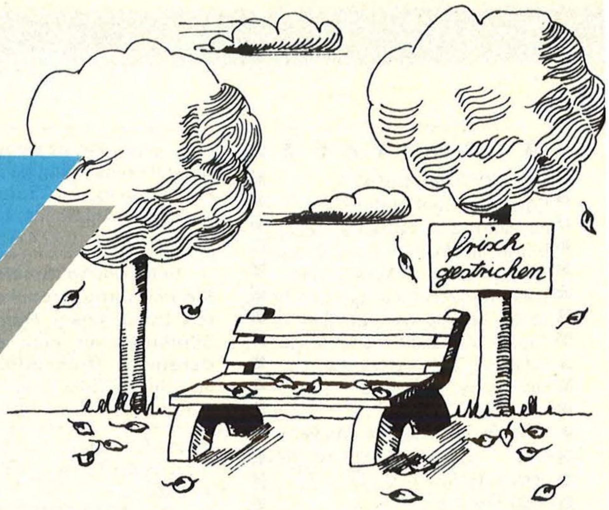
5,25" einseitig, 35 Spur, 50/10 Stck.	4,95/5,60
5,25" einseitig, 40 Spur, 50/10 Stck.	5,85/6,50
5,25" eins. 40 Sp., Verst.-Ring 50/10	6,28/6,98

Riesenauswahl an Spielen, Büchern und Zubehör in unserem Computer-Shop

MCPS Micro-Computer, Peripherie und Software GmbH
Verkauf: Gibitzenhofstraße 69, Postfach 14 21
8500 Nürnberg 1, Tel. 09 11/67 70 93
Versand per Nachnahme zuzüglich Postgebühren
Komplettpreisliste gegen 5,– DM in Briefmarken

Scan: S. Höltgen

Programmieren in BASIC



In diesem BASIC-Kurs werden vier der meistgekauften Home-Computer besonders berücksichtigt: Sinclair Spectrum, Commodore, Texas Instruments 99/4A und Atari

Bei der Farbgebung unterscheidet man auf dem Bildschirm drei Bereiche: Vordergrund, Hintergrund und Rahmen.

Sie entsprechen der Tinte, mit der man auf ein Blatt Papier schreibt, das in einem Rahmen eingespannt ist. Gemäß diesem Vergleich wurden die entsprechenden BASIC-Befehle beim Spectrum INK, PAPER und BORDER genannt.

Alle drei Bereiche können vom Home-Computer mit einer beliebigen Farbe aus seiner Palette belegt werden. Wenn es jedoch um die gute Lesbarkeit eines Textes geht, eignen sich nur bestimmte Farbkombinationen. Selbstverständlich sind gar keine Zeichen mehr zu erkennen, wenn Vordergrund- und Hintergrundfarbe übereinstimmen. Nach dem Einschalten des TI-99/4A könnte man der Meinung sein, daß dieser keinen Rahmen hat. Die Ursache hierfür liegt jedoch in der Standard-Einstellung transparent für den Hintergrund. Bei dieser „Farbe", die auch der Spectrum kennt, scheint die Farbe der dahinterliegenden Ebene (hier des Rahmens) hindurch.

Der VC 20 stellt speziell für den Rahmen acht zusätzliche Farben zur Verfügung (die wohl vom C 64 herübergeschwappt sein müssen), die im Vordergrund und Hintergrund nicht einsetzbar sind.

Wären die Gestaltungsmöglichkeiten damit erschöpft, dann könnte man jeweils nur maximal drei Farben gleichzeitig auf dem Bildschirm beobachten. Die Home-Computer bieten jedoch mehr. Man kann gewissermaßen die Tintenfarbe laufend wechseln, und auch der Hintergrund muß nicht eintönig bleiben. Ohne gewisse Einschränkungen würde jedoch der Bildschirmspeicher zuviel Platz beanspruchen.

Jeweils unterschiedlichen Speicherplatzbedarf haben die zwölf Grafikbetriebsarten des Atari (die hochauflösende Grafikbetriebsart 8 zum Beispiel belegt bis zu 8138 Byte). Sie unterscheiden sich in der Feinheit des Bildschirmrasters und der Anzahl der Farben und Helligkeitsstufen für die einzelnen Bereiche. Diese Vielfalt ist recht verwirrend, zumal die verwendeten Farbregister von Betriebsart zu Betriebsart ihre Zuständigkeit für einen bestimmten Bildschirmbereich wechseln können. Beim Spectrum sind die Verhältnisse überschaubarer: Er arbeitet immer hochauflösend und belegt für seinen Bildschirm 6912 Byte. 768 davon bestimmen die Vorder- und Hintergrundfarben sowie die Helligkeit und den Blinkzustand. 768, das ist die Anzahl 32 der Spalten multipliziert mit der Anzahl 24 der Zeilen; es ist also jedes Byte dieses Attribut-Speicher genannten Bereiches genau einem Feld des Zeilen-Spalten-

Code	Spectrum	VC 20	C 64	TI-99/4A	Atari
0	schwarz	schwarz	schwarz		grau
1	blau	weiß	weiß	transparent	goldfarben
2	rot	rot	rot	schwarz	orange
3	violett	türkis	türkis	mittelgrün	rot
4	grün	violett	violett	hellgrün	rosa
5	hellblau	grün	grün	dunkelblau	violett
6	gelb	blau	blau	hellblau	blau-purpur
7	weiß	gelb	gelb	dunkelrot	blau
8	transparent		orange	kornblumenblau	hellblau
9	Kontrast		braun	mittelrot	blau-grün
10			hellrot	hellrot	aqua
11			grau 1	dunkelgelb	grün-blau
12			grau 2	hellgelb	grün
13			hellgrün	dunkelgrün	gelb-grün
14			hellblau	magentarot	orange-grün
15			grau 3	grau	orange
16				weiß	
	+ jeweils zwei Helligkeitsstufen	+ acht Farben nur für den Hintergrund			+ jeweils acht Helligkeitsstufen

Tab. 23: Farbpaletten

Scan: S. Holtgen

Spectrum	VC 20	C 64	TI-99/4A	Atari
INK 8 PAPER 7 BORDER 7 BRIGHT 0 FLASH 0	PRINT CHR$(159); POKE 36879,27	PRINT CHR$(154); POKE 53281,6 POKE 53280,14	FOR I=1 TO 16 CALL COLOR(I,2,1) NEXT I CALL SCREEN(4)	GRAPHICS 0 SETCOLOR 0,2,8 SETCOLOR 1,12,10 SETCOLOR 2,9,4 SETCOLOR 3,4,6 SETCOLOR 4,0,0
schwarzer Vordergrund weißer Hintergrund weißer Rahmen normale Helligkeit kein Blinken	türkiser Vordergrund weißer Hintergrund türkiser Rahmen	hellblauer Vordergrund blauer Hintergrund hellblauer Rahmen	schwarzer Vordergrund transparenter Hintergrund hellgrüner Rahmen	aqua Vordergrund blauer Hintergrund schwarzer Rahmen

Tab. 24: Standardeinstellungen der Farben beim Einschalten

Spectrum	VC 20	C 64	TI-99/4A	Atari
			5 FOR I=1 TO 16 6 CALL CHAR (24+8*I, "FFFFFFFFFFFFFFFF") 7 CALL COLOR(I,I,1) 8 NEXT I	5 GRAPHICS 11
	10 DIM F(8)	10 DIM F(16)		
20 LET R=0	20 LET R=−64	20 LET R=−64	20 LET R=0	20 LET R=0
	30 FOR I=1 TO 8	30 FOR I=1 TO 16		
	40 READ F(I)	40 READ F(I)		
	50 NEXT I	50 NEXT I		
	60 DATA 144,5,28,159	60 DATA 144,5,28,159		
	70 DATA 156,30, 31,158	70 DATA 156,30, 31,158		
		80 DATA 129,149, 150,151		
		90 DATA 152,153, 154,155		
	100 PRINT CHR$(19);	100 PRINT CHR$(19);		
110 FOR I=0 TO 31	110 FOR I=1 TO 22	110 FOR I=1 TO 40	110 FOR I=1 TO 32	110 FOR I=0 TO 79
120 INK INT (RND*8+1)	120 PRINT CHR$(F(INT (RND(1)*8)+1));	120 PRINT CHR$ (F(INT(RND(1)*14) +1));	120 LET F=32+INT (RND*16)*8	120 COLOR INT(RND (1)*16)
130 FOR J=0 TO 21	130 FOR J=1 TO 22	130 FOR J=1 TO 24	130 FOR J=1 TO 24	130 FOR J=0 TO 191
140 PRINT AT J+ (21−2*J)*R,I;"■"	140 PRINT CHR$(18); " ";CHR$(81+R);	140 PRINT CHR$(18); " ";CHR$(81+R);	140 CALL HCHAR(J+ (25−2*J)*R,I,F,1)	140 PLOT I,J+ (191−2*J)*R
150 NEXT J	150 NEXT J	150 NEXT J	150 NEXT J	150 NEXT J
160 LET R=ABS (R−1)	160 LET R=R*(−1)	160 LET R=R*(−1)	160 LET R=ABS(R−1)	160 LET R=ABS(R−1)
	170 PRINT CHR$(29); CHR$(145);	170 PRINT CHR$(29); CHR$(145);		
180 NEXT I	180 NEXT I	180 NEXT I	180 NEXT I	180 NEXT I
190 GO TO 110	190 GOTO 100	190 GOTO 100	190 GOTO 110	190 GOTO 110

Tab. 25: Listings zum „Farbmuster"

Rasters zugeordnet; m. a. W. innerhalb des Feldes läßt sich der Füllfederhalter nicht absetzen, um ein Zeichen mit einer anderen Farbe zu beenden, und der Hintergrund des Feldes kann nur eine bestimmte Farbe annehmen oder diese komplett wechseln. Was beim Spectrum der Attribut-Speicher, ist beim Commodore-Rechner der Farbspeicher. Er ist nur für den Vordergrund jedes Rasterfeldes verantwortlich; Blinken und Helligkeitsstufen kennt er nicht. Der Hintergrund ist auf dem ganzen Bildschirm einheitlich gefärbt.

In jedem Byte dieses Farbspeichers sind nur drei bit (beim VC 20 mit acht Farben) beziehungsweise vier bit (beim C 64 mit 16 Farben) genutzt. Solche „Verschwendung" ist (das zeigt sich auch in anderem Zusammenhang) dem Spectrum völlig fremd. Beim TI-99/4A sind Farben nicht mit Bildschirmstellen verknüpft, sondern mit Teilen des Zeichensatzes zu je acht Zeichen. Ändert man die Zuordnung mit dem Unterprogramm COLOR, so sind alle Zeichen auf dem Bildschirm betroffen, die zu diesem Teil gehören.

Bei den Commodore-Rechnern kann die Farbwahl sowohl durch Poken des Speichers vorgenommen werden, als auch durch Farbsteuerzeichen in PRINT-Befehlen. Diesen Steuerzeichen sind Code-Zahlen zugeordnet, mit denen in Verbindung mit der Funktion CHR$ die Steuerzeichen ersetzt werden können. Diese Code-Zahlen stimmen jedoch nicht mit denjenigen beim Poken (Tabelle 23) überein. Wie bei den Commodore-Rechnern ist auch beim Spectrum das Poken des Attribut-Speichers nicht der Normalfall. Meistens werden die Befehle INK, PAPER und BORDER (BRIGHT und FLASH) benutzt, in Verbindung mit PRINT-Befehlen aber auch ohne sie.

Im nächsten HC-Heft beginnt ein Maschinensprachekurs für die Mikroprozessoren Z 80 und 6502 (6510).

Scan: S. Höltgen

2 = Laden einer bereits auf Kassette abgespeicherten Grafik
3 = Programmende

0020–0070
Eingabe der Parameter
a und b bestimmen die Form der Grafik (a muß immer negativ gewählt werden!)
d bestimmt die Anzahl der berechneten Punkte. Je größer d, desto runder der Plot, aber auch desto länger dauert seine Erstellung.
w bestimmt, wie eng die Linien nebeneinander liegen
k1, k2 und k3 legen die Farbe für INK, PAPER und BORDER fest
nr ist die (willkürliche) Nummer der Grafik

0075–0090
Möglichkeit, die Parameter zu ändern „j" oder „n"

0100–0195
Zeichnen des Rahmens und Beschriftung

0200
Skalierung der Grafik

0210–0250
Festlegen des ersten Poltpunktes

0500–0700
Berechnung und Zeichnung der Grafik; akustisches Signal am Plotende

0800–0830
Laden der bereits auf Kassette abgespeicherten Grafik

0840–0850
Möglichkeit einer Hardcopy des Plots „j" oder „n"

0900–0990
Hardcopy auf ZX Printer

9000–9030
Laden und Verifizeren des Programms „Bi-Graph"

9050–9150
Möglichkeit, den Bildschirminhalt auf Kassette abzuspeichern „j" oder „n"

9100–9150
Abspeichern des Bildschirminhalts auf Kassette

9998–9999
Programmende

TI 99/4A schreibt Briefe

Das Programm übernimmt den Aufbau eines Briefes und hilft beim Korrigieren des Textes. Als Drucker eignet sich der Epson MX100; der TI99/4A muß mit Extended-BASIC ausgestattet sein

Zu Beginn wählt man zwischen den beiden Schriftarten „normal" und „doppelt". Es werden nun die einzelnen Elemente des Briefkopfes abgefragt. Beim Absender kann auf gespeicherte Angaben zurückgegriffen werden.

Ein bereits vorhandener Text kann von Kassette eingelesen werden. Eine Textzeile ist in drei Segmente zu je 20 Zeichen unterteilt. Jedes Segment wird mit ENTER abgeschlossen. Eine Textzeile gilt als eingegeben, wenn man das dritte Segment mit ENTER abschließt.

Es sind folgende Steuer-Zeichenketten möglich.

„ffff" (Fehler) Es wird zur letzten Eingabe bzw. zum Zeilenanfang zurückgesprungen
„llll" Löschen Zeile
„eeee" Einfügen einer Zeile
„zzzz" Man trifft die Auswahl zwischen „Doppeldruck", „Einfachdruck" und „Zeile". Wählt man „Zeile" so kann man durch Eingabe einer Zeilennummer im Text umherspringen
„aaaa" (Aufhören) Die Briefeingabe wird beendet.

Die Buchstaben Ä, Ö, Ü, ä, ö, ü und ß sind über die Tasten [, ~,], {, |, }, ~ erreichbar. Die Taste @ steht für das Zeichen §.

Es gibt keine Fehlermeldung, wenn über das DIN-A4-Format hinausgeschrieben wird. Man kann den Text auf Kassette speichern.

Max Johann Lemberger

```
1000 REM ==========================================================================
1010 REM
1020 REM    MAX JOHANN LEMBERGER
1030 REM    LANDWIRTSCHAFTSMEISTER
1040 REM    HATTING 10
1050 REM    A-4922 ST. MARIENKIRCHEN/HAUSRUCK
1060 REM           --------------------------
1070 REM           öSTERREICH
1080 REM
1090 REM ==========================================================================
1100 REM
1110 REM    BRIEFSCHREIBEN
1120 REM
1130 REM    KONFIGURATION:
1140 REM    TI-99/4A+EXTENDED BASIC+RS232+CASS.RECORDER
1150 REM    EPSON MX100 II
1160 REM
1170 REM ==========================================================================
1180 REM
1190 REM
1200 OPEN #3:"RS232.BA=9600",VARIABLE 63
1210 OPEN #4:"RS232.BA=9600.LF",VARIABLE 63
1220 PRINT #3:CHR$(27);"8";! PAPIERENDEERKENNUNG AUS
1230 ON WARNING NEXT
1240 CALL CLEAR
1250 REM =====================   DEUTSCHE ZEICHENDEFINITION   =================
1260 CALL CHAR(64,"001C201824180438"):: CALL CHAR(91,"00441028447C4444"):: CALL
CHAR(92,"0044003844444438"):: CALL CHAR(93,"0044004444444438")
1270 CALL CHAR(123,"0000004410287C44"):: CALL CHAR(124,"0000004438444438"):: CAL
L CHAR(125,"0000004400444438"):: CALL CHAR(126,"0000003844484444")
1280 FOR I=0 TO 13 :: CALL COLOR(I,4,1):: NEXT I :: CALL COLOR(3,15,1):: CALL CO
LOR(4,15,1):: CALL SCREEN(2)
```

Scan: S. Höltgen


```
1290 CALL COLOR(13,16,16):: CALL COLOR(14,9,9)
1300 DIM E$(6),AB$(6),T$(6),Z$(82),SC(82),SC$(2),Y$(3),B$(4)
1310 UE$=""
1320 SC$(1)="DOPPEL" :: SC$(2)="NORMAL"
1330 PRINT TAB(8);"BRIEFSCHREIBEN": : :
1340 PRINT "MIT DOPPELSCHRIFT? j/n      j"
1350 ACCEPT AT(23,28)VALIDATE("jn")SIZE(-1):ST$
1360 IF ST$="n" THEN D,SCH=2 ELSE SCH,D=1
1370 DATA ==anrede==,==name==,==firma/beruf==,==strasse/nr.==,==postfach==,==pos
tl.z./post==,==land==
1380 DATA Max Johann Lemberger,Landwirtschaftsmeister,Hatting 10,,4922 ST.MARIEN
KIRCHEN/H.,öSTERREICH
1390 FOR I=0 TO 6 :: READ T$(I):: NEXT I
1400 FOR I=1 TO 6 :: READ E$(I):: NEXT I
1410 PRINT : : :
1420 PRINT "datum ?    - -"
1430 ACCEPT AT(23,9)SIZE(-10)VALIDATE(DIGIT,"-."):D$ :: IF LEN(D$)<8 THEN 1430
1440 DISPLAY AT(1,8)ERASE ALL:"ABSENDER"
1450 FOR I=1 TO 6
1460 DISPLAY AT(I*3,8):T$(I):: DISPLAY AT(1+I*3,1):E$(I)
1470 ACCEPT AT(1+I*3,1)SIZE(-24)BEEP:AB$(I):: IF POS(AB$(I),"ffff",1)<>0 THEN IF
 I>=2 THEN I=I-1 :: GOTO 1460
1480 IF I=5 THEN IF LEN(AB$(I))<8 THEN 1470
1490 IF I=1 OR I=3 THEN IF E$(I)=AB$(I)THEN 1510
1500 NEXT I
1510 FOR K=I TO 6 :: AB$(K)=E$(K):: NEXT K
1520 GOSUB 3050
1530 GOSUB 2870
1540 PRINT "TEXT AUF CASS. GESPEICHERT ?": :"          J/N              N"
1550 REM ======   VORHANDENEN TEXT VON CASSETTE EINLESEN   ====================
1560 ACCEPT AT(23,28)SIZE(-1)VALIDATE("JN"):ST$ :: IF ST$="N" THEN 1630
1570 OPEN #1:"CS1",INTERNAL,INPUT ,FIXED 192
1580 INPUT #1:ZZ
1590 FOR I=1 TO ZZ+1 STEP 2 :: INPUT #1:SC(I),Z$(I),SC(I+1),Z$(I+1):: NEXT I
1600 CLOSE #1
1610 PRINT : :"ÜBERARBEITEN  J/N          J"
1620 ACCEPT AT(23,28)SIZE(-1)VALIDATE("JN"):UE$ :: IF UE$="N" THEN UE$="J" :: GO
TO 2260
1630 ZL=1 :: CALL CLEAR
1640 REM ***********************************************************************
1650 REM                BRIEFTEXT EINGEBEN
1660 REM ***********************************************************************
1670 PRINT TAB(9);SEG$(Z$(ZL),1,20):: PRINT TAB(9);SEG$(Z$(ZL),21,20):: PRINT TA
B(9);SEG$(Z$(ZL),41,20): :
1680 FOR Z=ZL TO 80
1690 PRINT TAB(9);SEG$(Z$(Z+1),1,20):: PRINT TAB(9);SEG$(Z$(Z+1),21,20):: PRINT
TAB(9);SEG$(Z$(Z+1),41,20)
1700 IF INT(Z/2)<>Z/2 THEN CALL VCHAR(17,31,128,3):: CALL VCHAR(17,10,128,3)ELSE
 CALL VCHAR(17,31,136,3):: CALL VCHAR(17,10,136,3)
1710 CALL HCHAR(17,32,ASC(SEG$(STR$(Z),1,1))):: CALL HCHAR(18,32,ASC(SEG$(STR$(Z
)&" ",2,1)))
1720 IF SC(Z)>0 THEN SCH=SC(Z)
1730 DISPLAY AT(18,1)SIZE(7):SC$(SCH):: DISPLAY AT(22,1)SIZE(7):"ZL DR" :: DISPL
AY AT(24,1)SIZE(9):STR$(KO);TAB(3);KO+3 :: CALL SOUND(150,1400,1)
1740 REM ==================   EINGABE DER 3 20ER ZEILEN   =====================
1750 FOR Y=1 TO 3
1760 ACCEPT AT(16+Y,9)SIZE(-20):Y$(Y)
1770 REM =========   AN DEN ANFANG DER ZEILE ZURÜCKKEHREN   ===================
1780 IF POS(Y$(Y),"ffff",1)>0 THEN DISPLAY AT(16+Y,9)SIZE(20):" " :: CALL SOUND(
800,500,1):: GOTO 1720
1790 REM ===========   EINE ZEILE EINFüGEN   ==================================
1800 IF POS(Y$(Y),"eeee",1)=0 THEN 1830
1810 CALL SOUND(2000,1000,1):: FOR I=79 TO Z STEP -1 :: Z$(I+1)=Z$(I):: SC(I+1)=
SC(I):: Z$(I)="" :: NEXT I :: ZL=Z :: ZZ=ZZ+1 :: GOTO 1990
1820 REM =========*=   EINE ZEILE LöSCHEN   ===================================
1830 IF POS(Y$(Y),"llll",1)=0 THEN 1860
1840 CALL SOUND(2000,1500,1):: FOR I=Z TO 79 :: Z$(I)=Z$(I+1):: SC(I)=SC(I+1)::
Z$(I+1)="" :: NEXT I :: ZL=Z :: ZZ=ZZ-1 :: GOTO 1990
1850 REM =========   BRIEFEINGABE BEENDEN (AUFHöREN)   ========================
1860 IF POS(Y$(Y),"aaaa",1)<>0 THEN Z=Z-1 :: GOTO 2260
1870 REM ======   EINE VON DEN DREI ANSCHLIESZENDEN MöGLICHKEITEN WÄHLEN   =====
1880 IF POS(Y$(Y),"zzzz",1)=0 THEN 2030
1890 FOR I=1 TO 2 :: X=I*4 :: DISPLAY AT(X-3,1)SIZE(7):SC$(I):: DISPLAY AT(X-2,1
)SIZE(7):"SCHRIFT" :: DISPLAY AT(X-1,1)SIZE(7):I :: NEXT I
1900 DISPLAY AT(9,1)SIZE(7):"ANDERE" :: DISPLAY AT(10,1)SIZE(7):"ZEILE" :: DISPL
AY AT(11,1)SIZE(7):" 3"
1910 CALL SOUND(200,2000,1)
1920 CALL KEY(0,KE,STA):: IF STA=0 THEN 1920 ELSE CALL SOUND(200,2000,1)
1930 IF KE=49 THEN SCH=1 :: CALL HCHAR(3,5,136,4):: GOTO 1730
1940 IF KE=50 THEN SCH=2 :: CALL HCHAR(7,5,136,4):: GOTO 1730
1950 IF KE=51 THEN CALL HCHAR(11,5,136,4):: GOTO 1970
1960 GOTO 1920
1970 DISPLAY AT(13,1)SIZE(7):"ZEILE?"
1980 ACCEPT AT(14,1)VALIDATE(DIGIT)SIZE(7)BEEP:ZL :: IF ZL<1 OR ZL>80 THEN 1980
1990 IF UE$<>"J" AND ZL>Z THEN 1980
2000 Z$(0)="" :: PRINT TAB(9);SEG$(Z$(ZL-1),1,20):: PRINT TAB(9);SEG$(Z$(ZL-1),2
1,20):: PRINT TAB(9);SEG$(Z$(ZL-1),41,20): :
2010 GOTO 1670
2020 REM ====   20ER ZEILEN MIT LEERZEICHEN AUF 20 AUFFüLLEN   =================
2030 FOR I=LEN(Y$(Y))TO 19 :: Y$(Y)=Y$(Y)&" " :: NEXT I
2040 NEXT Y
2050 PRINT
2060 REM ======   ZEILE MIT NEUEM TEXT ÜBERNEHMEN   ===========================
2070 Z$(Z)=""
2080 FOR I=1 TO 3 :: Z$(Z)=Z$(Z)&Y$(I):: NEXT I
2090 REM ======   DOPPEL ODER EINFACHDRUCK ÜBERNEHMEN   ========================
2100 SC(Z)=SCH
2110 REM ======   ZEILENSPRUNG KONTROLLIEREN   =================================
2120 IF ZZ<Z THEN ZZ=Z
2130 REM ========   BEI DUPLIKAT DRUCKROUTINE AUSSCHALTEN   ====================
2140 IF UE$="J" THEN 2220
2150 REM =======   2 ZEILEN DRUCKABSTAND HALTEN   ==============================
2160 IF Z<KO+3 THEN 2220
2170 KO=Z-2
2180 REM ===========   DRUCKROUTINE ERSTMALIGE TEXTEINGABE   ===================
2190 ON SC(KO)GOTO 2200,2210
2200 PRINT #3:CHR$(27);"E";Z$(KO):: GOTO 2220 !DOPPELDRUCK EIN
2210 PRINT #3:CHR$(27);"F";Z$(KO)!DOPPELDRUCK AUS
2220 NEXT Z
2230 REM ***********************************************************************
2240 REM     BEI ERSTDRUCK LETZTE 2 ZEILEN - BEI ÜBERARBEITUNG GANZEN TEXT DRUCK
EN
2250 REM ***********************************************************************
2260 IF UE$="J" THEN KO=0
2270 FOR I=KO+1 TO ZZ
2280 IF SC(I)=1 THEN PRINT #3:CHR$(27);"E";Z$(I)ELSE PRINT #3:CHR$(27);"F";Z$(I)
2290 NEXT I
2300 CALL CLEAR
2310 REM ============================   UNTERZEICHNUNG   ===================
2320 C$(1)="Hochachtungsvoll" :: C$(2)="Mit freundlichen Grüßen" :: C$(3)="Mit f
reundlichem Gruß"
2330 DISPLAY AT(1,10)BEEP:"UNTERZEICHNUNG"
2340 FOR I=1 TO 3 :: DISPLAY AT(2+I*3,1):I;C$(I):: NEXT I
2350 DISPLAY AT(17,1):" 4 FREI DEFINIERT"
2360 CALL KEY(0,V,S):: IF S=0 THEN 2360
2370 V=V-48 :: IF V<1 OR V>4 THEN 2360
2380 IF V<4 THEN V$=C$(V):: GOTO 2400
2390 INPUT "Text? ":V$ :: IF LEN(V$)>20 THEN 2390
2400 CALL CLEAR :: PRINT V$
2410 REM =====================   EVTL. ANLAGE (BEILAGE) ANGEBEN   =============
2420 IF A$(1)<>"" THEN PRINT : :"ANLAGE GLEICH LASSEN j/n   j" ELSE GOTO 2450
2430 ACCEPT AT(23,28)SIZE(-1)VALIDATE("jn")BEEP:ST$
2440 IF ST$="j" THEN 2530
2450 FOR I=1 TO T :: A$(I)="" :: NEXT I
2460 DISPLAY AT(1,1):"ANLAGE? j/n                n"
2470 ACCEPT AT(1,28)SIZE(-1)VALIDATE("jn")BEEP:ST$ :: IF ST$="n" THEN I=0 :: GOT
O 2530
2480 FOR T=1 TO 7
2490 ACCEPT AT(2+T*2,1)SIZE(-28)BEEP:A$(T):: IF A$(T)="" THEN 2530
2500 IF POS(A$(T),"ffff",1)>0 THEN DISPLAY AT(2+T*2,1):" " :: CALL SOUND(800,500
,1):: IF T>0 THEN T=T-1 :: GOTO 2490
2510 NEXT T
2520 REM ======   ABSCHLUSS PLATZEINTEILUNG FüR DIN A4   =======================
2530 H=(36-ZZ-T)/2
2540 FOR I=2 TO H :: PRINT #3:" " :: NEXT I
2550 PRINT #3:TAB(28);V$
2560 IF ST$="n" THEN 2620
2570 FOR I=3 TO H :: PRINT #3:" " :: NEXT I
2580 PRINT #3:"ANLAGE": :
2590 FOR I=1 TO T
2600 PRINT #3:A$(I)
2610 NEXT *I
2620 CALL CLEAR :: CALL SOUND(300,400,5,500,5,600,5,-3,5):: PRINT "NEUES PAPIERB
LATT EINLEGEN !": : : :
2630 PRINT "briefkopie erwünscht j/n   j"
2640 ACCEPT AT(23,28)VALIDATE("jn")SIZE(-1)BEEP:TT$
2650 IF TT$<>"j" THEN 2690
2660 PRINT #3:CHR$(27);"F";:: GOSUB 2880
```

Scan: S. Höltgen


```
2670 FOR I=1 TO ZZ :: PRINT #3:Z$(I):: NEXT I
2680 GOTO 2540
2690 CALL CLEAR :: PRINT "AUF CASS. ABSPEICHERN? j/n n"
2700 ACCEPT AT(23,28)VALIDATE("jn")SIZE(-1)BEEP:ST$
2710 IF ST$="n" THEN 2770
2720 REM =========   TEXT AUF CASSETTE ABSPEICHERN   =========================
2730 OPEN #2:"CS2",INTERNAL,OUTPUT,FIXED 192
2740 PRINT #2:ZZ
2750 FOR I=1 TO ZZ+1 STEP 2 :: PRINT #2:SC(I),Z$(I),SC(I+1),Z$(I+1):: NEXT I
2760 CLOSE #2
2770 CALL CLEAR :: CALL SOUND(300,400,5,500,5,600,5,-3,5):: PRINT "NEUES PAPIERB
LATT EINLEGEN !": : : :
2780 PRINT "BRIEF MIT GLEICHEM ABSENDER WIEDERHOLEN  j/n           n"
2790 ACCEPT AT(23,28)SIZE(-1)VALIDATE("jn")BEEP:ST$
2800 IF ST$="j" THEN 2820
2810 GOSUB 3050
2820 SCH=0 :: GOSUB 2870
2830 GOTO 1610
2840 REM ********************************************************************
2850 REM       ABSENDER DRUCKEN
2860 REM ********************************************************************
2870 IF SCH=1 THEN PRINT #3:CHR$(27);"E";
2880 PRINT #3:AB$(1):AB$(2):AB$(3)
2890 IF AB$(4)<>"" THEN PRINT #3:AB$(4)
2900 PRINT #4:AB$(5)
2910 PRINT #3:"      ";RPT$("_",LEN(AB$(5))-5):TAB(7);AB$(6)
2920 PRINT #3:" ": :
2930 ED$=SEG$(AB$(3),1,POS(AB$(3)," ",1)-1)&", am "&D$
2940 PRINT #3:E$(0): :E$(1):E$(2):E$(3)
2950 IF LEN(E$(4))>1 THEN PRINT #3:E$(4)
2960 PRINT #4:E$(5):: IF E$(6)="" THEN PRINT #4:TAB(58-LEN(ED$));ED$
2970 PRINT #3:"      ";RPT$("_",LEN(E$(5))-5)
2980 IF E$(6)<>"" THEN PRINT #3:TAB(7);E$(6);TAB(58-LEN(ED$));ED$ ELSE PRINT #3:
" "
2990 IF E$(4)="" THEN PRINT #3:" "
3000 PRINT #3:" ": : : :" " :: PRINT #4:TI$ :: PRINT #3:RPT$("_",LEN(TI$)): : :
:
3010 RETURN
3020 REM ********************************************************************
3030 REM        EMPFÄNGER EINGEBEN
3040 REM ********************************************************************
3050 DISPLAY AT(1,8)ERASE ALL:"EMPFÄNGER"
3060 FOR I=0 TO 6
3070 DISPLAY AT(3+I*3,8):T$(I)
3080 ACCEPT AT(4+I*3,1)SIZE(-24)BEEP:E$(I):: IF POS(E$(I),"ffff",1)<>0 THEN IF I
>=1 THEN DISPLAY AT(4+I*3,1):" " :: I=I-1 :: GOTO 3080
3090 IF I=5 THEN IF LEN(E$(I))<8 THEN 3080
3100 NEXT I
3110 EE$=SEG$(E$(1),POS(E$(1)," ",1)+1,LEN(E$(1))-POS(E$(1)," ",1))&" !"
3120 REM ========================   ANREDE EMPFÄNGER   ========================
3130 DISPLAY AT(1,8)ERASE ALL:"ANREDE"
3140 B$(1)="Sehr geehrte Frau "&EE$ :: B$(2)="Sehr geehrter Herr "&EE$
3150 B$(3)="Geschätzte Damen und Herren* !" :: B$(4)="Sehr geehrte Damen und Herr
en !"
3160 FOR I=1 TO 4 :: DISPLAY AT(2+I*3,12):"==";STR$(I);"==":B$(I):: NEXT I
3170 DISPLAY AT(17,12):"==5==":"FREI DEFINIERT"
3180 DISPLAY AT(20,12):"==6==":"BETRIFFT: "
3190 CALL KEY(0,K,S):: IF S=0 THEN 3190
3200 K=K-48 :: IF K<1 OR K>6 THEN 3190
3210 CALL CLEAR
3220 IF K>4 THEN 3240
3230 TI$=B$(K):: GOTO 3280
3240 IF K=6 THEN PRINT "BETRIFFT: ": :
3250 CALL SOUND(200,500,1):: INPUT "TEXT ? ":TI$
3260 IF K=6 THEN TI$="BETRIFFT: "&TI$
3270 IF LEN(TI$)>60 THEN 3240
3280 CALL CLEAR :: PRINT TI$: : : :
3290 RETURN
```


Home-Computer

CREATIVISION 16 Farben, 6502A Mikroprozessor, 16 KByte Dynamic RAM. Erweiterung: Kassettenrekorder, Drucker, Floppy-Disc, 16 KByte RAM bzw. 64 KByte RAM und verschiedene Interface-Module. - 12 Spielkassetten. Weitere Software ist in Vorbereitung!

LASER 110/210 Mikroprozessor Z80A, 16 KByte ROM, 4 KByte RAM (LASER 210: 8 KByte RAM und 8 Farben), Tongenerator. Erweiterung: 16 KByte RAM bzw. 64 KByte RAM, Drucker, Interface-Modul. Zahlreiche Programme erhältlich sowie in Vorbereitung (z. B. Assembler)!

Bei Ihrem Händler oder bei Generalimporteur

SANYO

SANYO VIDEO Vertrieb GmbH & Co.
Lange Reihe 29, D-2000 Hamburg 1 - Tel. 040 / 24 62 66, Tx. 2 174 757

Scan: S. Höltgen

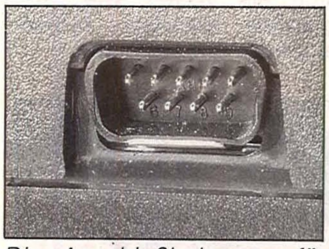
Für alle Fälle

Was tun, wenn die Bedienungsanleitung nicht weiterhilft? – An dieser Stelle erhalten Sie Tips für ganz alltägliche Situationen

Wie kann ich einen handelsüblichen Kassettenrecorder an den TI-99/4A anschließen?

Da gibt es verschiedene Lösungen. Zum Anschluß eines Kassettenrecorders ist beim TI-99/4A auf der linken Gehäuseseite eine 9polige Anschlußbuchse vorgesehen. Texas Instruments bietet ein Verbindungskabel an, das auf der einen Seite die drei Recorderstecker MIC, EAR und Remote besitzt und auf der anderen Seite genau in die Anschlußbuchse des Rechners paßt. Nachteil dieses Kabels: Es kostet im Handel rund 45 Mark.

Eine wesentlich billigere Lösung bietet sich an, wenn man sich die Einzelteile selbst besorgt und zusammenlötet. Der Aufwand: etwa eine halbe Stunde Arbeit und knappe drei Mark Materialkosten.



Die Anschlußbelegung für den Kassettenrecorder beim TI-99/4A:

Für den Kassettenrecorder müssen drei Klinkenstecker besorgt werden. Zwei gleiche für die Buchsen EAR und MIC sowie eine etwas dünnere für den Remote-Ausgang. Jeder dieser Klinkenstecker hat zwei Anschlüsse: Einer geht innen zur Spitze, und der zweite befindet sich am unteren Ende. Es müssen an jeden Klinkenstecker zwei Kabel angelötet werden. Am besten verwendet man Kabel in verschiedenen Farben, um spätere Verwechslungen zu vermeiden.

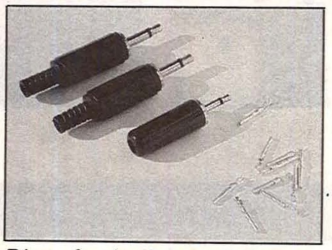
Für den Anschluß an den Rechner müssen jetzt an jedes dieser sechs Kabel kleine Kontaktstecker angelötet werden. Beim Kauf ist darauf zu achten, daß diese Stecker genau auf die Stifte am Recorderanschluß auf der Gehäuseseite passen.

Die neun Stifte am Rechneranschluß sind mit Nummern versehen. Die Kontaktstecker müssen auf folgende Stecker gesteckt werden:

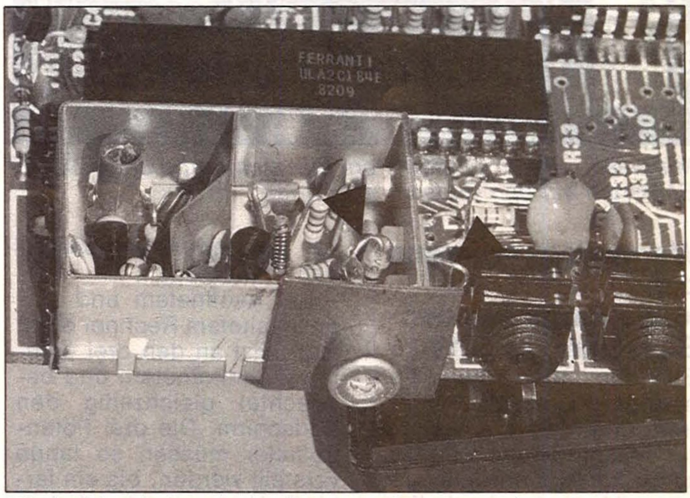
Stift	Klinkenstecker
5	MIC vorn
3	MIC hinten
2	Remote vorn
1	Remote hinten
9	EAR hinten
8	EAR vorn

Anschließend kann jeder handelsübliche Kassettenrecorder mit dem TI-99/4A betrieben werden.

Die Recorderstecker EAR, MIC und Remote sowie die Kontaktstecker beschafft man sich am besten in einem Fachgeschäft für elektronische Bauteile. Ansonsten dürfte es für jemanden, der mit einem Lötkolben umgehen kann, kein Problem sein, die Verbindung zu erstellen.



Die erforderlichen Teile kosten weniger als drei Mark



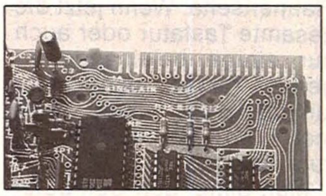
Die beiden Leitungen müssen getrennt werden. Anschließend wird das rechte Kabel von der Platine direkt mit der Antennenbuchse verbunden

Wie kann ich einen Monitor an den ZX81 anschließen?

Auf der Platine des ZX81 ist neben dem Fernsehausgang auch ein Monitorsignal vorhanden. Es ist allerdings von außen nicht zugänglich. Man kann allerdings die auf dem Foto gekennzeichneten Leitungen auftrennen. Das Kabel, das in den Fernsehmodulator hineinführt, muß direkt mit der Antennenbuchse verbunden werden. Anschließend kann am Fernsehausgang ein Monitor angeschlossen werden. Denkbar wäre auch, einen Schalter anzubringen, um zwischen Fernseher und Monitor umzuschalten.

Schwierigkeiten mit der Speichererweiterung beim ZX81

Zur Erhöhung der Speicherkapazität gibt es eine 16-KByte-Erweiterung für den Sinclair ZX81. Sie wird auf einer Steckerleiste an der Gehäuserückwand angebracht. Dadurch, daß diese Steckerleiste nicht geschützt ist, kann es zur Verunreinigung bzw. Oxidation der Kontakte kommen. Die Verbindung einzelner Signale vom Rechner zur Speichererweiterung ist unterbrochen. Es empfiehlt sich, vor Gebrauch des 16K-Moduls die Kontakte der Steckerleiste zu säubern. Sie befindet sich hinten rechts auf der Platine.



Vor dem Anschluß der Erweiterung sollten diese Kontakte gereinigt werden

Scan: S. Höltgen

Waterrun

In einem Labyrinth versperren Wasserbehälter den Weg. Nur durch geschicktes Umfüllen schlägt man sich zum Ausgang durch. Das Programm läuft auf der Grundversion des TI-99/4A

Nachdem der Rechner den Irrgarten auf den Bildschirm gezeichnet hat, stellt er leere und volle Wasserbehälter an zufällig ausgewählte Stellen. Die leeren sind kein unüberwindliches Hindernis: sie zerbrechen, wenn man gegen sie tritt. Sinnlose Zerstörung könnte sich später allerdings rächen! An einem vollen Wasserbehälter kommt man nämlich nicht vorbei, und umfüllen kann man diesen nur, wenn ein leerer erreichbar ist, der seinen Inhalt aufnimmt.
Die Fortbewegung im Irrgarten wird über die Richtungstasten „E", „S", „D" und „X" gesteuert. Man füllt um, indem man einen vollen Behälter berührt, zum leeren marschiert und diesen ebenfalls berührt. Da leere Behälter jeweils blau und volle braun gekennzeichnet sind, wechseln die beiden Behälter bei dieser Aktion natürlich ihre Farbe. In Ihrer Funktion als Wasserträger sind Sie ebenfalls blau. Ein Punktezähler steht zu Beginn bei tausend und wird bei jedem Ihrer Schritte um eins vermindert. Je längere Wege Sie gehen, um so weniger Punkte verbleiben Ihnen also beim Erreichen des Ausgangs.
Nach dem Programmstart werden Sie auf Wunsch noch einmal ausführlich über die Spielregeln informiert.

Michael Silberberg

```
100 REM  WATERRUN                    COPYRIGHT BY                 MICHAEL SILBERB
ERG          BERLINER STR.10              5657 HAAN 1                  **1983**
110 GOTO 330
120 CALL CLEAR
130 PRINT "SPIELREGELN ? (J/N)"::::::::::
140 CALL KEY(0,MS,ZX)
150 IF ZX=0 THEN 140
160 IF MS<>74 THEN 460
170 CALL CLEAR
180 PRINT TAB(10);"WATERRUN":::"SIE BEFINDEN SICH IN EINEM  LABYRINTH,DAS ZUSAET
ZLICH"
190 PRINT "NOCH DURCH WASSERBEHAELTER  VERSPERRT IST.(":"ALLERDINGS HABEN SIE
DIE    MOEGLICHKEIT,DIESE WASSERBE-HAELTER ZU LEEREN,INDEM"
200 PRINT "SIE DAS WASSER IN DIE LEERENBEHAELTER UMFUELLEN.()":"DURCH LEERE BEH
AELTER"
210 PRINT "KOENEN SIE DURCHLAUFEN,ZER- BRECHEN  DIESE ABER DABEI.":"SIE ENTLEERE
N bzw. FUELLEN  EINEN BEHAELTER,INDEM SIE"
220 PRINT "GEGEN EINEN SOLCHEN BE-      HAELTER RENNEN.":"SIE KOENNEN HOECHSTENS
EINENBEHAELTER AUF EINMAL ENT-"
230 PRINT "LEEREN.":"DRUECKEN SIE EINE TASTE.";
240 CALL KEY(0,MS,ZX)
250 IF ZX=0 THEN 240
260 CALL HCHAR(1,1,32,768)
270 PRINT "DIE STEUERUNG ERFOLGT UEBER DIE TASTEN 'E','S','D'&'X'  IN PFEILRICHT
UNG.":::"WENN SIE AUFGEBEN WOLLEN,   DANN BRAUCHEN SIE NUR  DIE  GLEICHHEITSTAST
E ZU";
280 PRINT " DRUECKEN.":::
290 PRINT "DRUECKEN SIE EINE TASTE."::
300 CALL KEY(0,MS,ZX)
310 IF ZX=0 THEN 300
320 RETURN
330 CALL CHAR(97,"FFFFFFFFFFFFFFFF")
340 CALL CHAR(140,"18183C5A99244242")
350 CALL COLOR(14,11,1)
360 WW=1000
370 CALL CHAR(155,"FF818181818181FF")
380 CALL CHAR(145,"FF818181818181FF")
390 CALL COLOR(16,5,8)
400 CALL COLOR(15,15,10)
410 CALL CLEAR
420 CALL SCREEN(16)
430 M$="LEER"
440 RANDOMIZE
450 GOSUB 120::GOSUB 1420
460 CALL CLEAR
470 PRINT "aEaaaaaaaaaaaaaaaaaaaaaaaaa"
480 PRINT "a      a    a a      a    a"
490 PRINT "a aaaaa   aaaa aaaaaa   aaaa"
500 PRINT "a a   aaa  a a a    aaa    a"
510 PRINT "a   aaa   aa   a aaaa   aa a"
520 PRINT "aa aa   aaa  a a a  a a a  a"
530 PRINT "a  a  a a   aaaa a aa a aaaa"
540 PRINT "aa a aa a aaa    a a  a    a"
550 PRINT "a    a  a a   aaaa a aa aaaa"
560 PRINT "a aa a aa aaa a a  a a    aa"
570 PRINT "aaa    a    a   aa   aaaa  a"
580 PRINT "a   aa    aa   aaa  aaa  aa a"
590 PRINT "a a a  aaa  aaa   aa aa a  a"
600 PRINT "a a a aa a aa a aaa  a  aaaa"
610 PRINT "aaa aaa  a a    a a aa aa  a"
620 PRINT "a   a   aa   aa a a a  a  aa"
630 PRINT "a a a a a  a a  a a a aa a a"
640 PRINT "aaa aaa a aa a aa   a a    a"
650 PRINT "a   a     a    a  a a   aaaa"
660 PRINT "aaaaa aaaaa aa   aa   aaa  a"
670 PRINT "a     a     a  aaa  aaa   aa"
680 PRINT "aaaaaaaaaaaaa aa          a aa"
690 PRINT "a             a  a a a a   a"
700 PRINT "aaaaaaaaaaaaaaaaaaaaaaaaaaAa";
710 FOR N=145 TO 155 STEP 10
720 FOR M=0 TO 30
730 X=INT(RND*25+3)
740 Y=INT(RND*22+2)
750 CALL GCHAR(Y,X,JO)
760 IF JO<>32 THEN 730
770 CALL HCHAR(Y,X,N)
780 CALL SOUND(-100,555+M,M)
790 NEXT M
800 NEXT N
810 X=4
820 CALL HCHAR(2,5,145)
830 CALL HCHAR(4,4,145)
840 Y=2
850 X1=X
860 Y1=Y
870 CALL HCHAR(Y,X,140)
880 IF (X=X1)*(Y=Y1)THEN 900
890 CALL HCHAR(Y1,X1,32)
900 CALL SOUND(-100,888-((24-Y)*3)-((28-X)*3),0)
910 Y1=Y
920 X1=X
930 CALL KEY(0,MS,ZX)
940 IF ZX=0 THEN 930
950 WW=WW-1
960 X=X-(MS=68)+(MS=83)
970 Y=Y-(MS=88)+(MS=69)
980 IF MS=61 THEN 1260
990 CALL GCHAR(Y,X,MS)
1000 IF MS=65 THEN 1210
1010 IF MS=32 THEN 1200
1020 IF MS=97 THEN 1180
1030 IF MS<>155 THEN 1100
1040 IF M$="VOLL" THEN 1180
1050 CALL COLOR(14,6,1)
1060 CALL HCHAR(Y,X,145)
1070 CALL SOUND(-100,-1,0)
1080 M$="VOLL"
1090 GOTO 1180
1100 IF MS<>145 THEN 1180
1110 IF M$<>"LEER" THEN 1140
1120 CALL SOUND(-100,-7,0)
1130 GOTO 1200
1140 CALL HCHAR(Y,X,155)
1150 M$="LEER"
1160 CALL COLOR(14,11,1)
1170 CALL SOUND(-100,-1,0)
1180 X=X1
1190 Y=Y1
1200 GOTO 870
1210 CALL HCHAR(1,1,32,768)
1220 FOR N=1 TO 10
1230 CALL SOUND(-100,RND*1000+110,RND*10)
1240 NEXT N
1250 IF WW>0 THEN 1270
1260 WW=0
1270 PRINT "BRAVO,SIE HABEN ES GE-      SCHAFFT.":::"IHRE PUNKTZAHL:";WW
1280 IF WW<QW THEN 1330
1290 FOR N=1 TO 10
1300 CALL SOUND(-100,110+N^3,N)
1310 NEXT N
1320 QW=WW
1330 PRINT :::"HIGH-SCORE:";QW
1340 PRINT ::"NOCH EIN SPIEL ? (J/N)"
1350 CALL KEY(0,MS,ZX)
1360 IF ZX=0 THEN 1350
1370 IF MS=ASC("N")THEN 1400
1380 WW=1000
1390 GOTO 330
1400 CALL CLEAR
1410 END
1420 CALL SOUND(4000,2000,0)
1430 RETURN
```

Scan: S. Höltgen

„Wurm" auf TI 99/4A

Auf einem Spielfeld versucht ein Wurm möglichst lange zu überleben. Futternäpfe stärken ihn, Giftnäpfe schwächen ihn.

Beim Leeren eines Napfes erscheinen jeweils drei neue, volle Näpfe. Das Spiel ist zu Ende, wenn der Wurm gegen den Zaun der Spielfeldbegrenzung rennt oder ihn alle Kraft verlassen hat. Der TI-99/4A teilt dann die erreichte Zeit und die Anzahl der geleerten Futter- bzw. Giftnäpfe mit.

Die Steuerung des Wurmes erfolgt mit einem Joystick. Mit folgenden Änderungen kann der Wurm auch über die Tastatur gesteuert werden:

```
1010 CALL KEY (0,K,S)
1020 IF S=0 THEN 1070
1030 IF K=101 THEN 1140
1040 IF K=120 THEN 1210
1050 IF K=115 THEN 1280
1060 IF K=100 THEN 1070
```

Einige Erklärungen zum Programm mit Angabe der Zeilennummern:

100–440 Anfangsroutine
450–550 Einstellen des Schwierigkeitsgrades
560–590 Aufbau des Spielfeldes
600–820 Definieren der Zeichencodes und der Variablen
830–890 Aufbau des Wurms
900–1000 Meldung über Berührung von Zaun oder Napf
1010–1430 Bewegen des Wurmes
1440–1600 Endroutine
1610–1760 Berührungserkennung
1770–1970 Setzen der neuen Näpfe
1980–2030 Einstellen des Schwierigkeitsgrades

Klaus Brackmann

```
100 CALL CLEAR
110 CALL SCREEN(14)
120 CALL COLOR(8,2,11)
130 CALL SOUND( 1000,110,0)
140 PRINT "          XXXXXXXX"
150 PRINT "          X      X"
160 PRINT "          X WURM X"
170 PRINT "          X      X"
180 PRINT "          XXXXXXXX"::::::::::::
190 PRINT "          co by k. brackmann"
200 FOR DELAY=1 TO 800
210 NEXT DELAY
220 CALL CLEAR
230 CALL SCREEN(4)
240 PRINT "kennst du die spielregeln?"
250 PRINT "<J><N>"
260 CALL KEY(0,K,S)
270 IF S=0 THEN 260
280 IF K=110 THEN 310
290 IF K=106 THEN 450
300 GOTO 240
310 PRINT "auf dem bildschirm erscheintein wurm und zwei futter-   naepfe.einer
der naepfe ent-haelt gift.":
320 PRINT "du musst nun mit hilfe einesjoystick versuchen,den rich-tigen napf zu
 treffen.":
330 PRINT "je oefter du triffst,je hoeher die lebenserwartung."::: 
340 PRINT "alles verstanden8<J><N>"
350 CALL KEY(0,K,S)
360 IF S=0 THEN 350
370 IF K=106 THEN 400
380 IF K=110 THEN 310
390 GOTO 310
400 CALL CLEAR
410 PRINT "gegen den zaun darfst du    natuerlich auch nicht laufen":::::::
420 PRINT "ES SIND AUCH LEERE          NAEPFE DA!"
430 FOR DELAY=1 TO 1500
440 NEXT DELAY
450 CALL CLEAR
460 PRINT "bist du neuling <N>?"::
470 PRINT "bist du fachmann <F>?"::
480 PRINT "bist du profi <P>?"::
490 CALL KEY(0,K,S)
500 IF S=0 THEN 490
510 IF K=110 THEN 1980
520 IF K=102 THEN 2000
530 IF K=112 THEN 2020
540 GOTO 460
550 CALL CLEAR
560 CALL HCHAR(2,2,88,28)
570 CALL HCHAR(22,2,88,28)
580 CALL VCHAR(3,2,88,19)
590 CALL VCHAR(3,29,88,19)
600 Q=8
610 R=22
620 M=12
630 T=9
640 CALL CHAR(144,"FF818181818181FF")
650 CALL CHAR(140,"FF818181818181FF")
660 GOSUB 1780
670 X1=10
680 X2=10
690 X3=10
700 X4=10
710 X5=10
720 XA=10
730 Y1=15
740 Y2=14
750 Y3=13
760 Y4=12
770 Y5=11
780 YA=10
790 P=0
800 G=0
810 F=0
820 GOTO 830
830 CALL VCHAR(X1,Y1,79)
840 CALL VCHAR(X2,Y2,79)
850 CALL VCHAR(X3,Y3,79)
860 CALL VCHAR(X4,Y4,79)
870 CALL VCHAR(X5,Y5,79)
880 CALL VCHAR(XA,YA,32)
890 CALL SOUND(-100,-4,0)
900 IF X1<3 THEN 1440
910 IF X1>21 THEN 1440
920 IF Y1<3 THEN 1440
930 IF Y1>28 THEN 1440
940 IF Q=X1 THEN 950 ELSE 960
950 GOSUB 1610
960 IF M=X1 THEN 980
970 GOTO 990
980 GOSUB 1640
990 P=P+1
1000 IF P=(V)THEN 1440
1010 REM
1020 CALL JOYST(1,X,Y)
1030 IF Y=4 THEN 1140
1040 IF Y=-4 THEN 1210
1050 IF X=-4 THEN 1280
1060 IF X=4 THEN 1070
1070 REM  NACH RECHTS
1080 XA=X5
1090 YA=Y5
1100 GOSUB 1350
1110 X1=X1
1120 Y1=Y1+1
1130 GOTO 830
1140 REM  NACH OBEN
1150 XA=X5
1160 YA=Y5
1170 GOSUB 1350
1180 X1=X1-1
1190 Y1=Y1
1200 GOTO 830
1210 REM  NACH UNTEN
1220 XA=X5
1230 YA=Y5
1240 GOSUB 1350
1250 X1=X1+1
1260 Y1=Y1
1270 GOTO 830
1280 REM  NACH LINKS
1290 XA=X5
1300 YA=Y5
1310 GOSUB 1350
1320 Y1=Y1-1
1330 X1=X1
1340 GOTO 830
1350 X5=X4
1360 X4=X3
1370 X3=X2
1380 X2=X1
1390 Y5=Y4
1400 Y4=Y3
1410 Y3=Y2
1420 Y2=Y1
1430 RETURN
1440 REM  END ROUTINE
1450 CALL SOUND(-400,-7,0)
1460 CALL CLEAR
1470 PRINT "DER WURM IST TOT"
1480 PRINT
1490 PRINT
1500 PRINT "FRESSNAPF: ";F;" MAL"
1510 PRINT "GIFTNAPF : ";G;" MAL"
1520 PRINT "LEBENSDAUER: ";P;" ZE"
1530 PRINT
1540 PRINT "NOCH EINMAL<J><N> ?"
1550 CALL KEY(0,K,S)
1560 IF S=0 THEN 1550
1570 IF K=106 THEN 450
1580 IF K=110 THEN 1600
1590 GOTO 1540
1600 END
1610 REM  BERUEHRUNG FRESSNAPF
1620 IF R=Y1 THEN 1630 ELSE 1670
1630 GOSUB 1680
1640 REM  BER.GIFTNAPF
1650 IF T=Y1 THEN 1660 ELSE 1670
1660 GOSUB 1730
1670 RETURN
1680 REM  FRESSNAPF
1690 F=F+1
1700 V=V+40
1710 CALL SOUND(-100,1000,0)
1720 GOTO 1780
1730 REM  GIFTNAPF
1740 G=G+1
1750 V=V-50
1760 CALL SOUND(-500,110,0)
1770 RANDOMIZE
1780 O=INT(RND*16)+1
1790 CALL COLOR(14,2,(O))
1800 CALL VCHAR(Q,R,32)
1810 RANDOMIZE
1820 FOR I=1 TO 3
1830 Q=INT((21-3+1)*RND)+3
1840 R=INT((28-3+1)*RND)+3
1850 CALL VCHAR(Q,R,140)
1860 NEXT I
1870 U=INT(RND*16)+1
1880 CALL COLOR(15,2,(U))
1890 CALL VCHAR(M,T,32)
1900 RANDOMIZE
1910 FOR I=1 TO 3
1920 M=INT((21-3+1)*RND)+3
1930 T=INT((28-3+1)*RND)+3
1940 CALL VCHAR(M,T,144)
1950 NEXT I
1960 RETURN
1970 STOP
1980 V=100
1990 GOTO 550
2000 V=70
2010 GOTO 550
2020 V=20
2030 GOTO 550
```

Scan: S. Höltgen

Für jeden etwas . . .

Wir haben uns sechs Spiele ausgesucht und für Sie getestet

Pac-Man

Ein kleines, süßes Wesen wohnt in einem Labyrinth. Als großer Fresser ist Pac-Man ständig auf der Suche nach Vitaminen und anderem Freßbaren, das im Labyrinth herumliegt. Doch da gibt es noch andere Labyrinthbewohner – die Monster. Sie trachten dem Pac-Man ständig nach dem Leben. Der Spieler muß Pac-Man so durch das Labyrinth steuern, daß er seinen großen Hunger stillen kann – für das Schlucken von Vitaminen et cetera gibt es Punkte – und daß er nicht von den Monstern gefressen wird. Aber Pac-Man kann auch selbst den Monstern an den Kragen – dafür gibt es Sonderpunkte –, wenn es ihm gelingt, eine der Kraftpillen zu schnappen.

Schwierigkeitsgrad: Acht Spielvarianten werden angeboten.

Spieler: 1 oder 2.

Unser Testurteil: Eines der spannendsten und interessantesten Spiele, das in letzter Zeit auf den Markt kam. Pac-Man ist ein gutes Beispiel dafür, wie Spielwitz und Spannung in ein Programm gepackt werden können, das sich nicht mit Weltraum und den Angriffen von Außerirdischen befaßt. Für Pac-Man kann es keine Altersempfehlung geben, das Spiel eignet sich aufgrund der einstellbaren Spielstufen für jeden.

Hersteller: Atari Elektronik.

Geeignet für: Atari 400/800 und 600 XL.

Preis: 149 Mark.

TI-Invaders

E.T., der Außerirdische, der sich auf die Erde verirrte, rührte die Kinogänger zu Tränen. Ein Schluchzen aus Hunderten von Kinogängerkehlen würden die TI-Invaders – Invasoren aus dem Weltraum – sicher nicht auslösen. Im Gegenteil. Wer diese widerlichen Kreaturen erblickt, denkt unwillkürlich an Verteidigung. So soll es auch sein. Die Invasoren greifen nämlich die Erde an. Ihr Kommandant hat ihnen befohlen, die Verteidiger trickreich auszumanövrieren. Sie wählen dafür zwei Angriffsarten – entweder sie feuern wahllos, ohne die Position der Raketenabschußrampen von den Verteidigern zu kennen, oder aber sie feuern gezielt auf die Abschußrampen. Der Verteidiger an der Abschußrampe des Computers muß Reaktionsschnelligkeit und Umsicht zeigen, um die vielfarbigen Kreaturen mit ihren schnellen Geschossen zu zerstören. Also: ständig in Stellung bringen, zielen und schießen.

Schwierigkeitsgrad: Verschiedene Variationen sind möglich.

Spieler: 1.

Unser Testurteil: Eines der klassischen Weltraumspiele. Geschossen wird, daß die Fetzen fliegen. Die Männchen, die hier die Eindringlinge aus dem Weltraum simulieren, sind in ihrer Form fast schon als „klassisch" zu bezeichnen. Was bringt das Spiel? Wir meinen, viel Action und Spannung.

Hersteller: Texas Instruments.

Geeignet für: TI 99/4 und 99/4A.

Preis: 79 Mark.

Demon Attack

Das hat noch gefehlt. Nach den Außerirdischen kommen jetzt die Weltraumvögel. Gott sei es gedankt – nur im Computer-Spiel. Und das vollzieht sich so: Die geflügelten Terroristen bombardieren den letzten Vorposten der Erde, die Mondstation Frieden. Gespenstisch, wie die Vögel auf die Verteidigungsstellung zukommen. Militärisch haben sie einiges drauf. Sonst würden sie sich nicht zu Staffeln formieren und in Wellen angreifen. Als Verteidiger am Computer weiß man aus der Spielbeschreibung, daß es vier Typen von Dämonen gibt: geflügelte Krieger, Terroristen mit Tentakeln, gefesselte Bomber und die Mitglieder der Selbstmordpatrouille. Für ihre militärische Ausrüstung verfügen sie gleich über drei verschiedene Typen von Bomben. Und was können die Verteidiger entgegenhalten? Laserkanonen mit hoher Treffgenauigkeit. Der Kampf kann beginnen. Aufgabe der Verteidiger am Bildschirm ist es, die Dämonen von der Landung auf der Erde abzuhalten.

Schwierigkeitsgrad: Mehrere Versionen sind möglich.

Spieler: 1 oder 2.

Unser Testurteil: Es ist ein sogenanntes Weltraumspiel, und damit mag es manchem engagierten Friedenskämpfer vielleicht als „Kriegsspiel" aufstoßen. Wir meinen jedoch, daß diese Parallele hier nicht gezogen werden muß. Die Programmierer haben viel Spannung in das Spiel gepackt. Aufregend wird es besonders dann, wenn die Angriffe unberechenbar vorgetragen werden.

Hersteller: Imagic.

Geeignet für: Atari 400/800 und Commodore VC 20 sowie C 64.

Scan: S. Höltgen

Tombstone City

Der Ort Tombstone City liegt irgendwo versteckt im Wilden Westen. Die Cowboys und Indianer haben sie längst verlassen. Eine Geisterstadt entsteht. Bis zu Spielbeginn gibt es in Tombstone City nur ein paar verlassene Häuser und jede Menge von Kaktusbäumen, ganz zu schweigen von viel Sand. Kaum gerät das Spiel in Bewegung, da wimmelt es nur so von grünen Monstern. Wo die herkommen, weiß der Geier. Der freut sich übrigens schon auf die Beute, die jetzt zwangsläufig geliefert werden muß. Die grünen Monster wollen nämlich die Stadt besetzen. Und das wieder wollen ein paar übriggebliebene Erdbewohner nicht. Nicht einmal ein paar Ruinen und Kaktusbäume gönnen diese Immobilienkapitalisten den Grünen – allerdings jenen vom anderen Stern. Nun gilt es also, die Monster abzuwehren. Und je mehr Punkte man dabei macht, um so größer wird die Bevölkerung von Tombstone City.

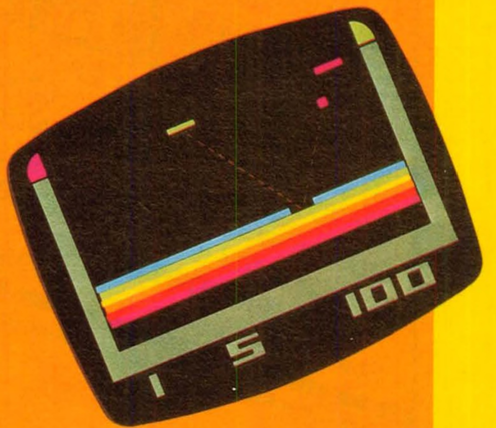
Schwierigkeitsgrad: Verschiedene Spielvariationen sind möglich.
Spieler: 1.
Unser Testurteil: Kein leichtes Spiel. Man muß sich zuerst einmal durch die Anleitung kämpfen. Wenn allerdings der Ablauf durchschaut ist, dann wird es spannend. Auch Kinder können mit diesem Spiel begeistert werden. Die verschiedenen Spielvariationen sind so gehalten, daß jeder eine Abwechslung hat. Wie bei Texas Instruments üblich, wird nur das Spielmodul in einen Kassettenschlitz des Computers geschoben.
Hersteller: Texas Instruments.
Geeignet für: TI 99/4, TI 99/4A.
Preis: 99 Mark.

Munch-Man

Die lieben Tierchen, die da in einem Labyrinth über den Bildschirm zokkeln, sind nicht so genau auszumachen. Das sei hier vorweggestellt, damit niemand in Sorge kommt, vielleicht im Biologieunterricht etwas versäumt zu haben. Selbst Tierärzte lüften das Geheimnis nicht. Wer hat schließlich schon einmal etwas von Hoonos gehört? Hoonos heißen also die Viecher, die da auf dem Bildschirm zu sehen sind. Schlau sind sie, das muß ihnen zugestanden werden. Und mit den Hoonos bekommt es auch der Spieler zu tun. Ihm ist nämlich die Aufgabe gestellt, Ketten zu bilden, ohne daß sie von den Hoonos aufgefressen werden. Gelingt das nicht, dann ist der Munch-Man dran. Den fressen die Biester auf, wenn sie nicht durch das Ziehen von Ketten daran gehindert werden. Damit die Sache nicht zu einfach wird, läuft das Geschehen mit vielen Wenn und Aber ab. Das Spiel ist so angelegt, daß es schnell verstanden werden kann und auch keine geistigen Klimmzüge erfordert.
Schwierigkeitsgrad: Verschiedene Spielvarianten sind möglich.
Spieler: 1.
Unser Testurteil: Das Spiel läßt sich schnell durchschauen, womit auch die Spannung nicht allzu lange anhalten dürfte. Gefallen hat uns die übersichtliche Darstellung des Geschehens auf dem Bildschirm sowie die gute farbliche Wiedergabe. Texas Instruments kam mit Munch-Man schon ziemlich früh auf den Markt. Im Vergleich mit später konzipierten Spielen erinnert es stark an den großen Bestseller der letzten Jahre – Pac-Man. Und dabei drängen sich selbstverständlich Vergleiche auf. Pac-Man hat nach unserer Meinung mehr spannende Details eingebaut, leidet aber dadurch bei verschiedenen Herstellern – es gibt mehrere Versionen – an der Übersichtlichkeit. Und die ist bei Munch-Man eindeutig besser.
Hersteller: Texas Instruments.
Geeignet für: TI 99/4, TI 99/4A.
Preis: 128 Mark.

Super-Breakout

Super-Breakout stellt den Ausbruch aus einem Gefängnis dar: Ein Kugelball muß auf dem Bildschirm so geschickt bewegt werden, daß er aus einer Mauer möglichst viele Ziegelsteine schlägt. So leicht, wie sich das anhört, ist die Sache auch wieder nicht. Man glaubt gar nicht, wie oft der Kugelball sein Ziel verfehlt und nicht genau das Loch in der Mauer weiter aushöhlt. Der Schwierigkeitsgrad läßt sich steigern, so daß die Sache noch interessanter wird.
Schwierigkeitsgrad: Fünf Versionen stehen zur Auswahl.
Spieler: 1 oder 2.
Unser Testurteil: „Viel Spannung und Unterhaltung in einem Spiel, das nicht im entferntesten nach Weltraum riecht." Das Programm ist so konzipiert, daß auch Kinder ihre Freude daran haben und vor allem keinen Schaden daran nehmen. Doch werden auch Erwachsene und fortgeschrittene Super-Breakout-



Spieler nicht alleine gelassen. Sie werden bei dem Schwierigkeitsgrad „Progressive" ihren Spaß haben.
Hersteller: Atari Elektronik.
Geeignet für: Atari 400/800 und 600 XL.
Preis: 139 Mark.

Scan: S. Höltgen

Schatzsuche

Monster, Magier und Gespenster wollen Sie am Entdecken eines Schatzes hindern. Zur Verteidigung läßt der TI-99/4A die Wahl zwischen einem Buch mit Zauberflüchen, einem Schwert und einem Ring

Haben Sie zur Verteidigung einen Ring gewählt, so können Sie durch Wände gehen. Haben Sie das Buch mit Zauberflüchen gewählt, so können Sie während der Schatzsuche gegen Ihre Gegner einen speziellen Fluch ausstoßen (nicht gegen den Magier). Der Fluch wirkt nur über kurze Entfernung und hat zur Folge, daß sich der Gegner in Luft auflöst.

Der Magier ist immun gegen Zauberflüche; wenn er nah genug an Sie herankommt, kann er Sie wegzaubern. Das Gespenst kann durch Wände gehen. Das Monster hat einen harten Schuppenpanzer und ist deshalb unempfindlich gegen Schläge des Schwertes.

Die Bewegungen können über die Tasten E, S, D und X jeweils in Pfeilrichtung vorgenommen werden.

Michael Silberberg

```
100 REM  SCHATZSUCHE                  COPYRIGHT BY                      MICHAEL SILBERB
ERG          BERLINER STR.10                5657 HAAN 1                       **1983**
110 GOTO 360
120 CALL CLEAR
130 PRINT "            SCHATZSUCHE":::
140 PRINT "SIE SUCHEN IN EINER HOEHLE  EINEN SCHATZ.":"UNTERWEGS MUESSEN SIE GEG
EN MONSTER,MAGIER UND"
150 PRINT "GESPENSTER ANKAEMPFEN.":"IHNEN STEHEN DREI ARTEN DER VERTEIDIGUNG ZUR
 VERFUEHGUNG..................."
160 PRINT :::"DRUECKEN SIE EINE TASTE."
170 CALL KEY(0,MS,ZX)
180 IF ZX=0 THEN 170
190 CALL CLEAR
200 PRINT "ZEICHENERKLAERUNG:"::"d =MAGIER (IMMUN GEGEN       ZAUBERFLUECHE,WENN
ER NAH"
210 PRINT "GENUG AN SIE HERAN KOMMT,    KANN ER SIE WEGZAUBERN)"::
220 PRINT " =GESPENST (KANN DURCH DIE WAENDE GEHEN)"::"
 =MONSTER (IMMUN GEGEN"
225 REM     GESPENST=CTRL+B              MONSTER=CTRL+L                   MAENNCHEN=CTR
L+>
230 PRINT "SCHLAEGE DES SCHWERTES,DA ESEINEN HARTEN SCHUPPENPANZER BESITZT)"
240 PRINT ::"=SIE SELBST."::"} =SCHATZ,DEN SIE FINDEN      SOLLEN."::"DRUECKEN S
IE EINE TASTE."
250 CALL KEY(0,MS,ZX)
260 IF ZX=0 THEN 250
270 CALL CLEAR
280 PRINT "WENN SIE AM UNTEREN BILD-    SCHIRMRAND ANKEGOMMEN SIND, DANN WIRD ERS
T EINMAL EIN"
290 PRINT "NEUES TEILSTUECK DER HOEHLE AUSGEDRUCKT."::
300 PRINT "SIE BEWEGEN IHRE FIGUR UEBERDIE TASTEN E,S,D&X IN PFEIL-RICHTUNG."::
310 PRINT "DRUECKEN SIE EINE TASTE."
320 PRINT ::::
330 CALL KEY(0,MS,ZX)
340 IF ZX=0 THEN 330
350 GOTO 550
360 CALL CHAR(97,"FFFFFFFFFFFFFFFF")
370 CALL CHAR(125,"3C7EE3EFE7F7463C")
380 DIM M$(18)
390 CALL SCREEN(4)
400 CALL COLOR(16,16,1)
410 CALL COLOR(15,15,1)
420 CALL COLOR(12,14,1)
430 CALL COLOR(14,5,1)
440 CALL COLOR(13,7,1)
450 CALL CHAR(100,"185A7E1818244242")
460 CALL CHAR(130,"00183C3C5A66FF7E")
470 CALL CHAR(140,"3C5A663C5A5A99A5")
480 RANDOMIZE
490 CALL CLEAR
500 CALL CHAR(155,"18187E5A183C2424")
510 PRINT "SIND DIE SPIELREGELN NOETIG?(J/N)":::::::
520 CALL KEY(0,MS,ZX)
530 IF ZX=0 THEN 520
540 IF MS=74 THEN 120
550 PRINT "GEBEN SIE EIN,WAS SIE ALS    AUSRUESTUNG MITNEHMEN WOLLEN............
..............."
560 WW=2000
570 PRINT ::::"WAEHLEN SIE UNTER":"EINEM BUCH MIT ZAUBER-       FLUECHEN":"EINEM
ZAUBERSCHWERT":
580 PRINT "ODER EINEM RING,DER IHNEN    ERMOEGLICHT DURCH WAENDE ZU GEHEN":::
590 INPUT "GEBEN SIE BITTE JETZT EINE   DER DREI MOEGLICHKEITEN EIN (BUCH,SCHWERT
,RING)":A$
600 IF (A$<>"BUCH")*(A$<>"SCHWERT")*(A$<>"RING")THEN 590
610 IF A$<>"BUCH" THEN 760
620 CALL CLEAR
630 PRINT "FUER JEDEN ANGREIFER GIBT ESEINEN SPEZIELLEN ZAUBERFLUCH............
..............":
640 PRINT :"GEBEN SIE IM SPIELVERLAUF   JEWEILS DEN ANFANGSBUCH-     STABEN EIN,G
EGEN DEN DER"
650 PRINT "FLUCH GERICHTET IST,ALSO ZB.(G) FUER EINEN ANTI-GESPENSTFLUCH usw."
660 PRINT "DER FLUCH WIRKT NUR UEBER    EINE KURZE ENTFERNUNG!"
670 PRINT "GEGEN DEN SCHWARZEN MAGIER   KANN KEIN FLUCH GERICHTET    WERDEN."
680 PRINT ::"DER FLUCH BEWIRKT,DASS SICH DER GEGNER IN LUFT AUFLOEST."::
690 WAF=1
700 PRINT "DRUECKE EINE TASTE."
```

```
710 CALL KEY(0,MS,ZX)
720 IF ZX=0 THEN 710
730 CALL CLEAR
740 CALL SOUND(100,440,0)
750 CALL SOUND(1000,444,0)
760 IF A$<>"SCHWERT" THEN 870
770 PRINT :::::"SIE BEDIENEN IHR SCHWERT     UEBER DIE TASTEN 'A' , 'F'  UND UEB
ER DIE 'SPACE'-TASTE."
780 PRINT "WENN SIE TASTE 'A' DRUECKEN,DANN BEWEGT SICH DAS SCHWERTNACH LINKS.BE
FINDET SICH"
790 PRINT "ETWAS AUF DER POSITION AUF  DIE DAS SCHWERT AUFTRIFFT,   DANN WIRD DER
 GEGENSTAND      ZERSTOERT."
800 PRINT "MIT 'SPACE' SCHLAGEN SIE AUFDIE STELLE VOR IHNEN."
810 PRINT "MIT 'F' SCHLAGEN SIE NACH     RECHTS.IHR SCHWERT WIRKT NURDANN,WENN SIE
 EINE DIESER    TASTEN DRUECKEN."
820 PRINT "DRUECKEN SIE EINE TASTE."
830 CALL KEY(0,MS,ZX)
840 IF ZX=0 THEN 830
850 WAF=2
860 CALL CLEAR
870 IF A$<>"RING" THEN 940
880 CALL CLEAR
890 PRINT "DIE EINZIGE WIRKUNG DES      RINGES BESTEHT DARIN,DASS    SIE DURCH WAE
NDE GEHEN        KOENNEN.":::::::
900 WAF=3
910 PRINT "DRUECKEN SIE EINE TASTE."
920 CALL KEY(0,MS,ZX)
930 IF ZX=0 THEN 920
940 CALL CLEAR
```

```
950 DATA "a    a      a       a      a","a                  a       a","a a
               a   a"
960 DATA "a     a               a    a","a       a                  a","a   a a
a a            a   a"
970 DATA "a                         aa","a a           aa           a","a
                   a"
980 DATA "a               a a        a","a    a            a        a","a
         a   a     a"
990 DATA "a                 a        a","a a   a a   a              a","a a
        a          a"
1000 DATA "a      a     a    a    a   a","a          a          a    a","a   a a
       a a         aa"
1010 RESTORE 950
1020 FOR N=1 TO 18
1030 READ M$(N)
1040 NEXT N
1050 FOR BH=1 TO 10
1060 Y1=2
1070 X1=15
1080 CALL CLEAR
1090 A=24
1100 B=INT(RND*20+5)
1110 A1=A
1120 B1=B
1130 PRINT "aaaaaaaaaaaaaaaaaaaaaaaaaaaa"
1140 FOR N=1 TO 22
1150 PRINT M$(INT(RND*18+1))
1160 NEXT N
1170 XZ=INT(RND*20+6)
1180 ON INT(RND*3+1)GOTO 1190,1210,1230
1190 GEG=100
1200 GOTO 1240
1210 GEG=130
1220 GOTO 1240
1230 GEG=140
1240 Y=2
1250 X=15
1260 CALL HCHAR(Y,X,155)
1270 IF BH<10 THEN 1290
1280 CALL SOUND(100,444,0)
1290 WW=WW-1
1300 IF (Y=Y1)*(X=X1)THEN 1320
1310 CALL HCHAR(Y1,X1,32)
1320 IF GEG=32 THEN 1360
1330 CALL HCHAR(A,B,GEG)
1340 IF (A=A1)*(B=B1)THEN 1360
1350 CALL HCHAR(A1,B1,32)
1360 CALL KEY(0,MS,ZX)
1370 X1=X
1380 Y1=Y
1390 IF BH<>10 THEN 1410
1400 CALL HCHAR(20,XZ,125)
1410 Y=Y-(MS=88)+(MS=69)-(Y<2)
1420 X=X-(MS=68)+(MS=83)-(X<3)+(X>30)
1430 IF Y<23 THEN 1480
1440 IF BH<10 THEN 1470
1450 Y=22
1460 GOTO 1480
1470 NEXT BH
1480 CALL GCHAR(Y,X,ZX)
1490 IF ZX=125 THEN 2270
1500 IF (ZX<>97)+(WAF=3)THEN 1530
1510 Y=Y1
1520 X=X1
1530 IF (ZX=100)+(ZX=130)+(ZX=140)THEN 2130
1540 ON POS("A FGM",CHR$(ABS(MS)),1)+1 GOTO 1920,1550,1650,1720,1820,1890
1550 IF WAF<>2 THEN 1920
1560 CALL CHAR(145,"0502050B102040")
1570 CALL GCHAR(Y+1,X-1,ZX)
1580 CALL HCHAR(Y+1,X-1,145)
1590 FOR V=1 TO 30
1600 NEXT V
1610 CALL HCHAR(Y+1,X-1,32)
1620 IF (ZX=130)+(ZX=100)THEN 1630 ELSE 1920
1630 GEG=32
1640 GOTO 1920
1650 CALL CHAR(145,"183C1818181818")
1660 CALL GCHAR(Y+1,X,ZX)
1670 CALL HCHAR(Y+1,X,145)
1680 FOR V=1 TO 30
1690 NEXT V
1700 CALL HCHAR(Y+1,X,32)
1710 GOTO 1620
1720 IF WAF<>2 THEN 1920
1730 CALL CHAR(145,"A040A0100B0402")
1740 CALL GCHAR(Y+1,X+1,ZX)
1750 CALL HCHAR(Y+1,X+1,145)
1760 FOR V=1 TO 30
1770 NEXT V
1780 CALL HCHAR(Y+1,X+1,32)
1790 IF (ZX=130)+(ZX=100)THEN 1800 ELSE 1920
1800 GEG=32
1810 GOTO 1920
1820 IF WAF<>1 THEN 1920
1830 IF GEG<>130 THEN 1920
1840 ZX=ABS(Y-A)+ABS(X-B)
1850 IF ZX>3 THEN 1920
1860 GEG=32
1870 CALL HCHAR(A,B,32)
1880 GOTO 1920
1890 IF WAF<>1 THEN 1920
1900 IF GEG<>140 THEN 1920
1910 GOTO 1840
1920 A1=A
1930 B1=B
1940 IF GEG=32 THEN 1260
1950 A=A+((RND>RND)*(SGN(A-Y)))-(A<2)
1960 IF GEG=130 THEN 2000
1970 CALL GCHAR(A,B,ZX)
1980 IF ZX<>97 THEN 2000
1990 A=A1
2000 B=B-(X>B)+(B>X)-(B<3)+(B>30)
2010 IF X<>B1 THEN 2030
2020 B=B-(RND>RND)+(RND>RND)
2030 CALL GCHAR(A,B,ZX)
2040 IF GEG=130 THEN 2070
2050 IF ZX<>97 THEN 2070
2060 B=B1
2070 IF (A=Y)*(B=X)THEN 2130
2080 IF GEG=100 THEN 2100
2090 GOTO 1260
2100 IF RND*2>RND THEN 1260
2110 ZX=ABS(X-B)+ABS(A-Y)
2120 IF ZX>3 THEN 1260
2130 IF GEG<>100 THEN 2160
2140 PRINT :::"DER MAGIER HAT SIE          VERZAUBERT.":::
2150 GOTO 2200
2160 IF GEG<>140 THEN 2190
2170 PRINT :::"DAS MONSTER HAT SIE          GEFRESSEN.":::
2180 GOTO 2200
2190 PRINT ::::"DAS GESPENST HAT SIE ZU TODEERSCHRECKT.":::
2200 PRINT ::::::"WOLLEN SIE NOCHMAL SPIELEN ?(J/N)"::::::
2210 CALL KEY(0,MS,ZX)
2220 IF ZX<=0 THEN 2210
2230 IF MS=74 THEN 550
2240 IF MS<>78 THEN 2210
2250 CALL CLEAR
2260 END
2270 PRINT ::"BRAVO,SIE HABEN ES GESCHAFFTDEN SCHATZ ZU FINDEN."::"IHRE PUNKTZAH
L:";WW
2280 IF WW<QW THEN 2330
2290 QW=WW
2300 FOR N=1 TO 10
2310 CALL SOUND(-100,RND*1000+110,0)
2320 NEXT N
2330 PRINT "HIGH-SCORE:";QW
2340 GOTO 2200
```


Scan: S. Höltgen

Scan: S. Höltgen

Die Super-Sound-Box

Eine Kombination von einigen integrierten Schaltkreisen und durch kleine Programme ergeben sich zahlreiche Soundvariationen

Viele Besitzer von Klein-Computern können bereits Musik per BASIC erzeugen, jedoch der Griff zu Lötkolben und Experimentierplatine zeigt neue Möglichkeiten für einen Computersound.

Als Rechteckgenerator verwenden wir den Baustein 555, der sich in einem achtpoligen DIL-Gehäuse befindet. Der Vorteil dieses Bausteines ist nicht nur der sehr günstige Preis von etwa 1,50 Mark, sondern auch die sehr einfache Beschaltung. Außerdem darf bei diesem Baustein die Betriebsspannung zwischen 4 V und 15 V schwanken, ohne daß sich die Funktionsweise ändert.

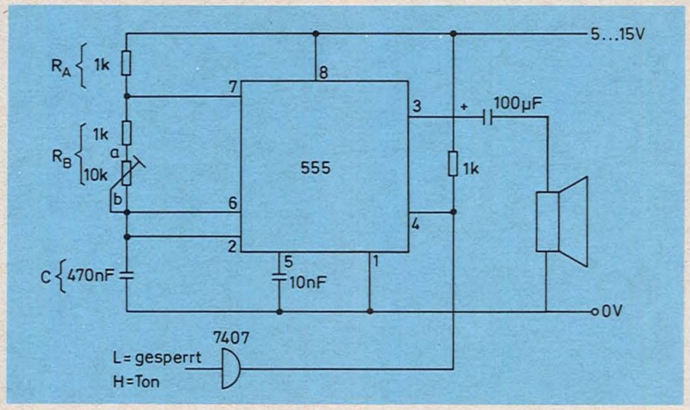
Der Mini-Sound

An Pin 1 schließen wir die Masse von unserem Computer an. Bei der Betriebsspannung von Pin 8 können wir entweder die interne Stromversorgung verwenden, oder wir schließen ein externes Netzgerät mit 12 V oder 15 V an, was auf jeden Fall besser ist.

kann sich der Kondensator mit 470 nF nach einer e-Funktion aufladen, wobei der Aufladestrom über die beiden Widerstände mit je 1 kΩ fließt.

Im Betrieb

Erreicht die Kondensatorspannung den Wert von ⅔ der Betriebsspannung, ändert der 555 sein Verhalten. Bei einer Betriebsspannung von 12 V liegt diese Umschaltschwelle bei 8 V und wird von dem Eingang U_{schw} gemessen. Der interne Transistor von Pin 7 (U_{ent} = Entladung) wird leitend, und der Kondensator entlädt sich über den 1 kΩ-Widerstand. Die Entladung erfolgt nach einer e-Funktion (exponentiell), bis die Spannung den Wert von ⅓ der Betriebsspannung, beispielsweise 4 V, unterschreitet. Über Pin 2 mit der Bezeichnung Trigger (Tr) verändert der Baustein 555 sein Verhalten und sperrt den internen Transistor von Pin 7. Nun kann sich der Kondensator wieder nach einer e-Funktion aufladen.



Einfacher Rechteckgenerator mit Schnittstelle zum Computer

Die Ladezeit für den Baustein 555 errechnet sich aus

$t_1 = 0{,}7 \cdot (R_A + R_B) \cdot C$

Bei einem Widerstand von R_A = 1 kΩ erhalten wir eine Zeit von 65,8 ms. Stellen wir den Trimmer auf 10 kΩ ein, errechnet sich eine Ladezeit von 3,95 ms. Die Entladezeit bestimmt jedoch nur der Widerstand R_B und

$t_2 = 0{,}7 \cdot R_B \cdot C$

Es ergeben sich die Zeiten von 33 ms (1 kΩ) und 3,6 ms (10 kΩ + 1 kΩ). Die Periodendauer für die Frequenzerzeugung ist

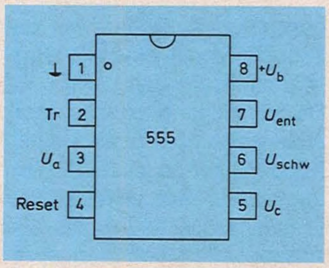
$T = 0{,}7\,(R_A + 2 \cdot R_B) \cdot C$

und die Ausgangsfrequenz errechnet sich aus

$$f = \frac{1}{T} = \frac{1}{0{,}7\,(R_A + 2 \cdot R_B) \cdot C}$$

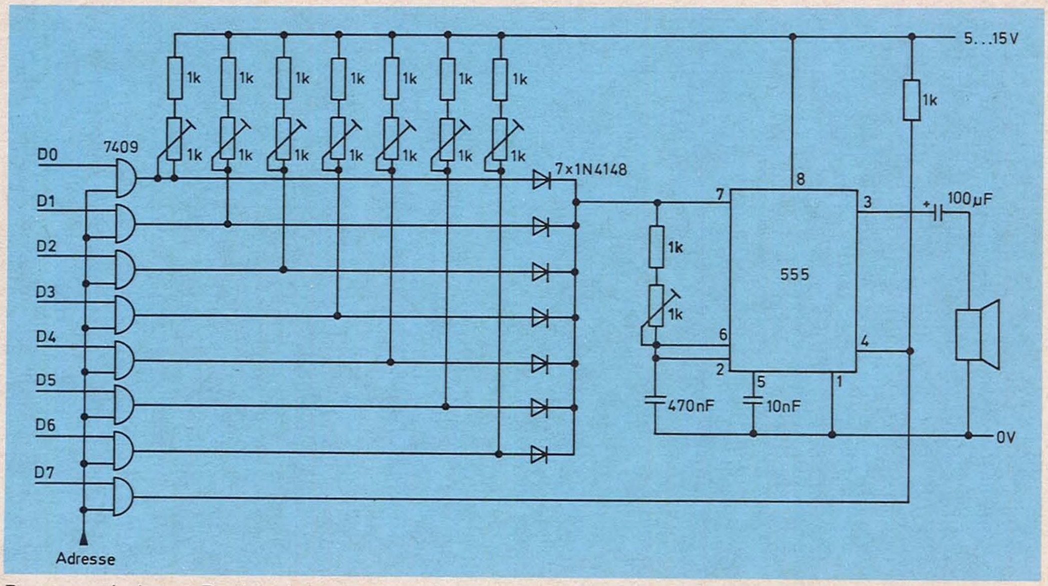
Für unsere Schaltung läßt sich durch den Trimmer die Frequenz zwischen 1520 Hz und 130 Hz stufenlos einstellen.

Als Schnittstelle zwischen Mini-Sound-Box und Computer setzen wir einen Puffer vom Typ 7407 ein. Hat der Reset-Eingang (Pin 4) einen L-Pegel, kann der 555 nicht arbeiten.



Anschlußschema des Timers 555, der als Rechteckgenerator arbeitet

Bei dem Mini-Sound erzeugt uns der Timer 555 nur eine Ausgangsfrequenz, die wir aber einstellen können. Legen wir den Schleifer von dem Trimmer auf a, hat dieser keinen ohmschen Wert. Schalten wir die Betriebsspannung ein,



Programmierbarer Rechteckgenerator mit 128 verschiedenen Tonfolgen

Scan: S. Höltgen

Bei einem H-Pegel nimmt der 555 sofort seine Arbeit auf und ein Ausgangston wird erzeugt.

Die Schaltung läßt sich für die Mini-Sound-Box einfach realisieren. Das Programm arbeitet mit
POKE XXXXX,XX

Nach dem POKE-Befehl folgt die Adresse und der Wert. Wir müssen den Eingang von dem 7407 entweder auf L- oder H-Pegel legen. Der 555 gibt so lange einen Ton ab, bis der Puffer wieder einen L-Pegel hat.

Die erweiterte Mini-Sound-Box

Durch die Steuerung der Ladezeiten für den Timer 555 durch den Computer ergeben sich bis zu 128 verschiedene Töne und Tonfolgen.

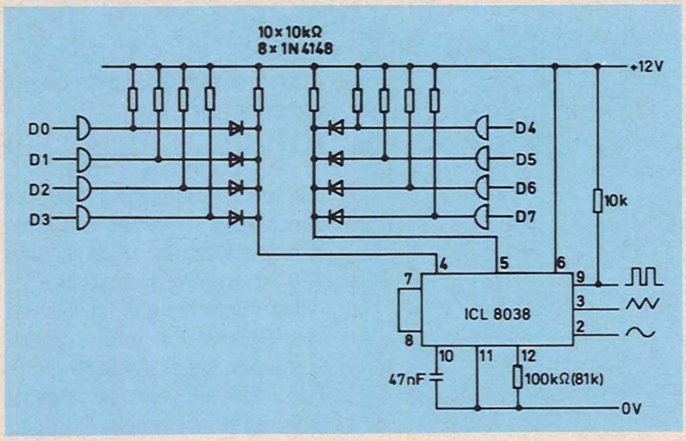
Wir steuern die Aufladezeit des Kondensators durch unterschiedliche Widerstandskombinationen an. Dabei müssen wir jedoch Dioden als elektronische Schalter für die Entkopplung der einzelnen Zweige einschalten. Die Schnittstelle zwischen Computer und Sound-Box sind acht UND-Gatter vom Typ 7409. Dieser Baustein beinhaltet jeweils vier UND-Gatter mit offenem Kollektor.

Hat die Datenleitung D_0 und die Adressenfreigabe einen H-Pegel, ist die UND-Bedingung erfüllt und der Ausgangstransistor vom offenen Kollektor gesperrt. Über den Festwiderstand und den Trimmer kann für den Timer 555 ein Aufladestrom fließen, wobei wir die Schleusenspannung der Diode noch berücksichtigen müssen. Durch den Abgleich mittels der einzelnen Trimmer erhalten wir für jede Ansteuerungswertigkeit, die nach dem dualen Zahlensystem erfolgt, eine andere Ausgangsfrequenz. Liegen zwei Ausgänge auf H-Pegel, kann sich der Kondensator über zwei Widerstände aufladen, und die Ladezeit verkürzt sich entsprechend. Die Ausgangsfrequenz wird höher.

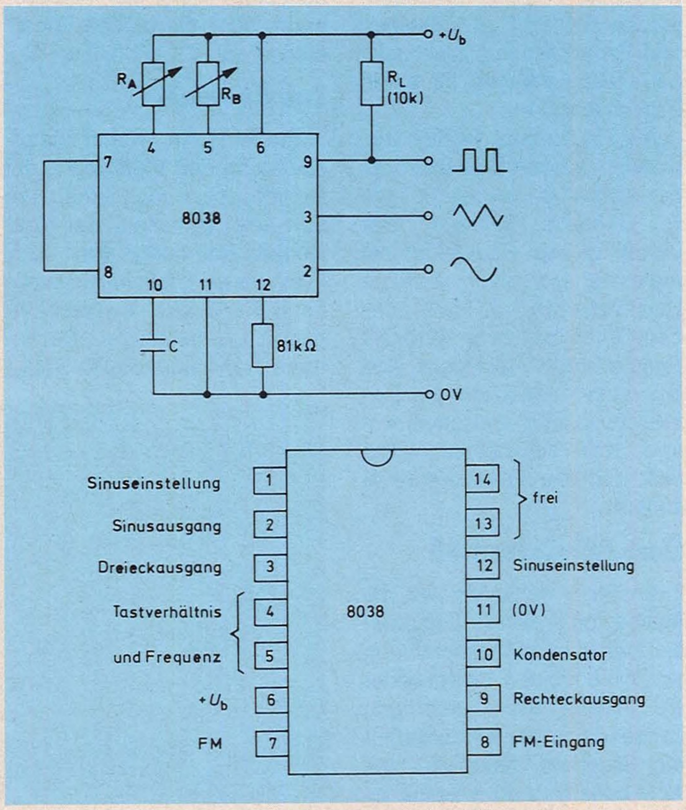
Insgesamt stehen uns für die Programmierung des Timers 555 sieben Eingänge zur Verfügung, das heißt, wir können $2^7 = 128$ Töne erzeugen. Die Datenleitung D7 benötigen wir für die Freigabe. Zwischen Computer und den UND-Gattern muß sich ein Zwischenspeicher befinden, damit der Wert für die Tonfolge gespeichert wird.

Etwas Orgeleffekt

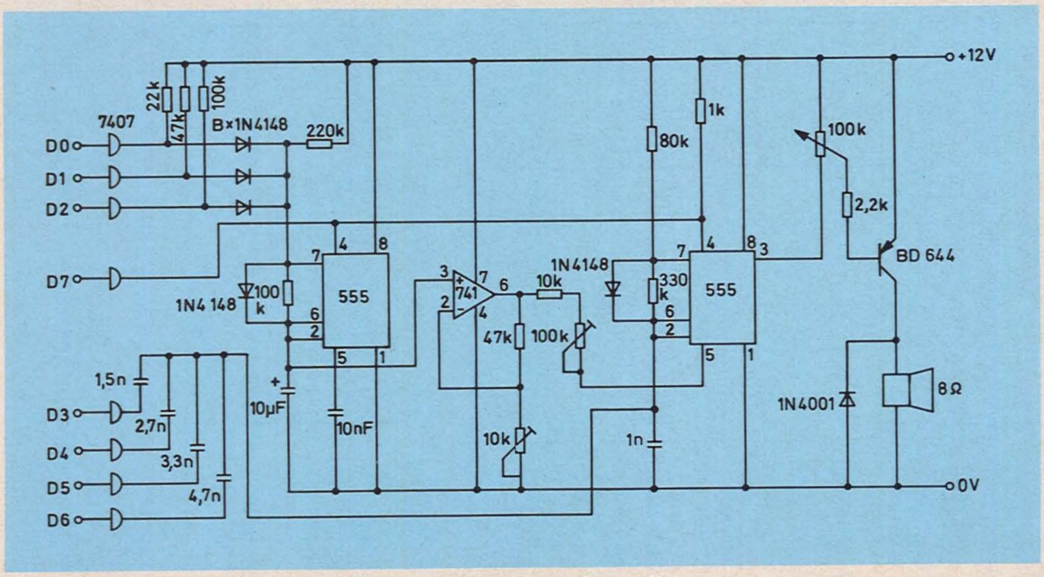
Der Timer 555 erzeugt nicht nur eine Rechteckfrequenz am Ausgang (Pin 3), sondern auch eine Dreieck- oder Sägezahnfunktion an Pin 2 und Pin 6. Dies kommt von den Lade- und Entladezeiten des Kondensators. Außerdem hat der Timer 555 noch einen Kontrollspannungseingang (Pin 5). Diesen Eingang legen wir normalerweise über einen kleinen Kondensator an Masse. Schalten wir hier eine veränderliche Frequenz auf, ändert sich die Ausgangsfrequenz entsprechend. Steigt die Spannung an Pin 5, erhöht sich die Ausgangsfrequenz, verringert sich die Spannung, erhalten wir eine niedrigere Ausgangsfrequenz.



Schaltung einer Juke-Box mit drei Ausgangsfunktionen



Anschlußschema und Grundschaltung des ICL 8038



Schaltung für die Maxi-Sound-Box mit einem kleinen Orgeleffekt

Die Maxi-Sound-Box

Am Eingang der Schaltung befinden sich zwei 7407 mit offenem Kollektor. Dadurch

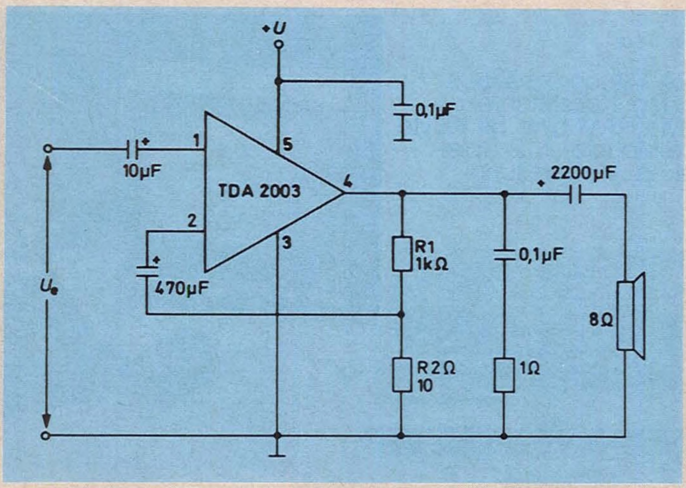
Scan: S. Höltgen

können für die beiden Timer die Steuerfunktionen erzeugt werden. Der linke Timer arbeitet als Orgelfunktion, der rechte 555 als Rechteckgenerator. Zwischen den beiden Schaltkreisen befindet sich der Operationsverstärker 741, der die Dreieckspannung verstärkt.

Mit den Datenleitungen D0, D1 und D2 bestimmen wir die Ladezeit von dem 10µF-Kondensator. Für die Ladefunktion kann der 100-kΩ-Widerstand nicht arbeiten, da parallel dazu eine Diode liegt. Die Aufladezeit wird praktisch nur von den drei Widerständen bestimmt, die von den Gattern angesteuert werden. Wir erhalten 2^3 = acht Tonfolgen, wobei sich durch den 220 kΩ eine Grundfolge ergibt. Der Kondensator entlädt sich über den 100-kΩ-Widerstand, da die Diode nun in Sperrichtung betrieben wird.

Die Funktionen

Die Dreieck- oder Sägezahnspannung liegt direkt an dem nichtinvertierenden Eingang des Operationsverstärkers. Die Einstellung der Ausgangsspannung nehmen wir über den 10-kΩ-Trimmer vor. Die Verstärkung läßt sich in einem weiten Bereich einstellen. Ebenfalls läßt sich durch den 100-kΩ-Trimmer die Spannung für den Eingang von Pin 5 beeinflussen. Wir können praktisch alle Funktionen nach unseren Wünschen justieren.



Leistungsverstärker mit HiFi-Qualitäten für die Sound-Box

Die eigentliche Ausgangsfrequenz bestimmen wir durch fünf Kondensatoren. Als Grundwert setzen wir 1 nF ein und schalten über den Computer vier weitere Kondensatoren hinzu. Die Parallelschaltung erhöht die Kapazität, und die Frequenz verringert sich. Da vier Dateneingänge in der Schaltung vorhanden sind, ergeben sich 2^4 = 16 Tonfolgen. Die Schaltung kann jedoch nur dann arbeiten, wenn der Computer den Eingang D7 auf H-Pegel legt. Hat dieser Eingang einen L-Pegel, können beide Timer nicht arbeiten.

Am Ausgang der Schaltung befindet sich ein Leistungstransistor, der im A-Betrieb arbeitet. Die Diode am Lautsprecher verhindert eine Selbstinduktion und der Transistor ist geschützt.

Die Juke-Box

Statt dem Timer 555 können wir auch den integrierten Funktionsgenerator ICL 8038 verwenden, der uns drei Ausgangsfunktionen erzeugt: Sinus, Rechteck und Dreieck. Durch die Ansteuerung des Computers ergeben sich zahlreiche Klangvariationen.

Der Bereich der Ausgangsfrequenz liegt zwischen 0,1 Hz und 50 kHz. Höhere Frequenzen lassen sich nicht erzielen, wie die Versuche zeigten. Die Ausgangsfrequenz wird von den beiden Widerständen R_A und R_B bestimmt. Wählen wir zwei Werte mit je 10 kΩ und einen mit 470 nF, erhalten wir 64 Hz an den drei Ausgängen.

Über den Eingang FM läßt sich der Baustein wobbeln, das heißt, die Ausgangsfrequenz erhöhen oder verringern. Wir verbinden Pin 7 und Pin 8, damit sich keine Wobbelfunktion ergibt. An Pin 8 läßt sich aber auch der Ausgang des Operationsverstärkers von der Maxi-Sound-Box anschließen. Die Ausgangsfrequenz ändert sich um einen Wobbel- oder Frequenzhub von etwa 50 zu 1. Dies bedeutet, daß die Ausgangsgrundfrequenz von 64 Hz auf 3,2 kHz durchgefahren wird.

Die einzelnen Ausgangsfrequenzen müssen wir noch verstärken. Hier eignet sich der TDA 2003 am besten. Bei einer Betriebsspannung von 12 V gibt der Baustein eine Leistung von 8 W ab, vorausgesetzt, der Baustein hat einen Kühlkörper, damit die Wärme schnell abgeführt wird. Ohne ausreichenden Kühlkörper spricht der interne Überlastungsschutz an. Damit ist der TDA 2003 gut geschützt.

Herbert Bernstein

Spielgeräte

Ein Joystick und ein Lichtgriffel für den VC 20 bedeuten optimale Spielmöglichkeiten, wobei die Hard- und Software besonders einfach aufzubauen beziehungsweise zu programmieren sind

Über einen Joystick läßt sich ein Ball auf dem Fernsehschirm in X- und Y-Richtung bewegen. Setzen wir einen Lichtgriffel ein, können wir auf dem Fernsehschirm Bilder und Figuren zeichnen.

Für die Realisierung von Joystick und Lichtgriffel befindet sich auf der rechten Seite des Computer-Gehäuses ein „Control-Port“ mit einem 9poligen Stecker. Die Pinbelegung lautet:

Pin	Funktion
1	Joystick 0
2	Joystick 1
3	Joystick 2
4	Joystick 3
5	Potentiometer Y
6	Lichtgriffel
7	+5 V/10 mA
8	0 V
9	Potentiometer X

Für den Anschluß an den VC 20 benötigen wir einen

Scan: S. Höltgen

Für alle Fälle

Was tun, wenn die Bedienungsanleitung nicht weiterhilft? – An dieser Stelle erhalten Sie Tips für ganz alltägliche Situationen

Print Using für Atari

Mit dem folgenden Programm haben Sie die Möglichkeit, auf den Atari-Computern Zahlen formatiert auszugeben.

```
        Programm für die formatierte Ausgabe von Zahlen

1 DIM Q$(15)
32750 REM Unterprogramm fuer die
32751 REM formatierte Ausgabe von
32752 REM Zahlen xxx (c) by TT83
32753 REM
32754 REM ATARI-Club Thomas Tausend
32755 REM    Am Felsenkeller 15
32756 REM    8764 Kleinheubach
32757 REM   Tel.: (09371)/4647
32758 REM
32759 REM  (c) 1983 by Thomas 1000
32760 REM  Version 1.0 - 02.11.83
32761 REM
32762 Q$="               ":Q=INT(Qx100+0.5)/100:Q$(LEN(Q$)-LEN(STR$(Q)),LEN(Q$))=STR$(Q)
32763 IF Q$(LEN(Q$)-3,LEN(Q$)-3)="." THEN RETURN
32764 IF Q$(LEN(Q$)-2,LEN(Q$)-2)="." THEN Q$(1,LEN(Q$))=Q$(2,LEN(Q$)-1):RETURN
32765 Q$(1,LEN(Q$)-4)=Q$(4,LEN(Q$)):Q$(LEN(Q$)-3,LEN(Q$))=".00"
32766 RETURN


Demo für obiges Programm:


10 REM xxx DEMOPROGRAMM FUER DIE xxx
20 REM xx FORMATIERTE AUSGABE VON xx
30 REM xxx  ZAHLEN  (c) by TT83  xxx
40 REM
50 Z=RND(0)x10000-5000
60 Q=Z
70 GOSUB 32750
80 PRINT Z,Q$
90 GOTO 50
```

Ermittlungen des freien Speicherplatzes bei Texas Instruments

Im TI-BASIC gibt es keinen Befehl, mit dem man den zur Verfügung stehenden Speicherplatz ermitteln kann. Es muß ein kleiner Umweg eingeschlagen werden. Man gibt, nachdem das zu überprüfende Programm geladen ist, folgende Programmzeilen ein:

1 X=X+8
2 GOSUB 1

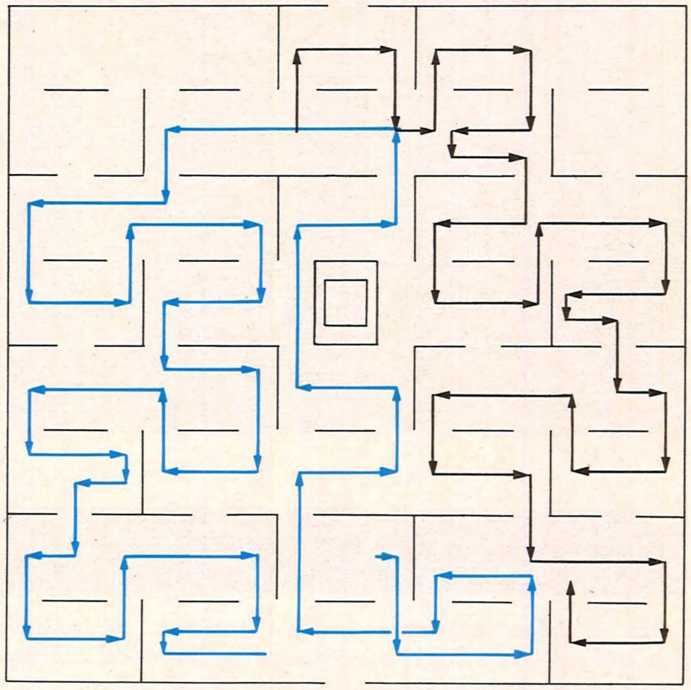
Nun startet man das Programm mit RUN. Nach einiger Zeit erscheint am Bildschirm die Fehlermeldung:
MEMORY FULL IN 1.

Jetzt wird im Direktmodus, also ohne Zeilennummer, eingegeben:
PRINT "FREI ="; X + 40; "BELEGT ="; 14573-X.

Der Rechner gibt jetzt sowohl die Anzahl der noch freien als auch der belegten Speicherplätze aus. Bevor nun mit dem ursprünglichen Programm weitergearbeitet wird, muß man die Zeilen 1 und 2 wieder entfernen. Falls X im Programm schon verwendet wurde, muß eine andere Variable gewählt werden.

Tricks für Pac Man

Bereits in der Novemberausgabe haben wir einige Hinweise gegeben, wie man beim Spiel Pac-Man einen möglichst gefahrlosen Weg durch das Labyrinth gehen kann. Hier nun ein Vorschlag, um den Gespenstern Blinky, Inky, Pinky und Clyde – wie sie bei Atari heißen – zu entkommen. Falls Sie ähnliche Wege herausgefunden haben, senden Sie uns diese bitte zu. Wir werden sie gesammelt in einer der nächsten Ausgaben veröffentlichen.



Bessere Kühlung für ZX 81

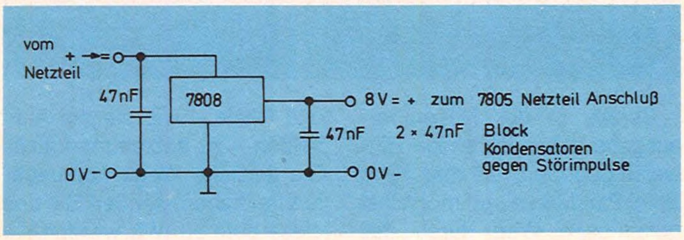
In der Ausgabe 11 brachten wir bei den Lesertips Hinweise, um die übermäßige Erhitzung des ZX 81 zu vermindern. Hierzu nun eine Ergänzung von unserem Leser Georg Naß aus Bremen.

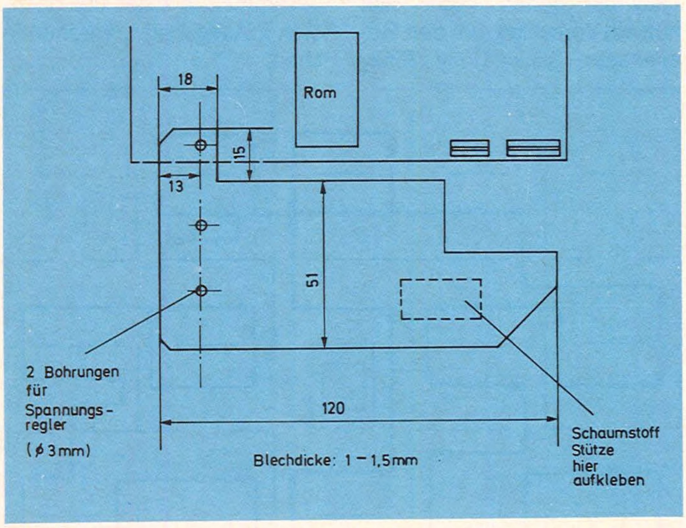
Wenn man die flachen Gummifüße des ZX 81 wegen der besseren Luftzirkulation gegen größere auswechselt, sollte man vor dem Aufkleben die Gummifüße in der Mitte mit einer Zange lochen. Dies hat den Vorteil, daß man beim späteren Öffnen des Rechners die Schrauben lösen kann, ohne die Gummifüße entfernen zu müssen.

Auch mit einem vergrößerten Kühlblech läßt sich die Wärmeentwicklung reduzie-

Scan: S. Höltgen

ren. Am besten verwendet man dazu etwa 1 bis 1,5 mm dickes Kupferblech. Das freischwebende Ende des Kühlblechs sollte durch ein Stück Schaumstoff zum Gehäuse hin abgestützt werden, damit die Platine durch das erhöhte Gewicht nicht belastet wird. Um die Erwärmung des eingebauten Spannungsreglers 7805 zu reduzieren, besteht weiterhin die Möglichkeit, über einen zweiten vorgeschalteten Spannungsregler mit der Bezeichnung 7808 die Spannung von 10 oder mehr Volt auf 8 Volt zu reduzieren. Der zweite Spannungsregler läßt sich leicht auf dem vergrößerten Kühlblech anbringen, wenn man die zwei angegebenen Bohrungen benützt. Durch das vergrößerte Kühlblech erhält der ZX 81 außerdem eine verbesserte Standfestigkeit.





Programmiertricks für ZX 81

Die vielfältigen Möglichkeiten der Programmierung erlauben auch bei einem so kleinen Computer wie dem ZX 81 Funktionen zu erledigen, die sonst nur professionellen Rechnern vorbehalten sind. Will man z. B. erreichen, daß bei der Ausgabe verschiedenstelliger Zahlen der Dezimalpunkt jedesmal an derselben Stelle steht – unabhängig davon, wie lange die Zahl ist –, genügt bereits die Eingabe der Programmzeile:

PRINT TAB X-LEN STR$ INT (Z); Z

X bedeutet dabei die Position des Dezimalpunktes in der entsprechenden Bildschirmspalte. Die Zahl selbst ist in der Variablen Z gespeichert.

Zu beachten ist, daß die Anzahl der Stellen vor dem Komma den Wert X nicht überschreiten darf. Falls die Zahl als Zeichenkette vorliegt, muß die Programmzeile in

PRINT TAB X-LEN STR$ INT VAL Z$; Z$

abgeändert werden.

Besonders nützlich ist diese Funktion beim Schreiben von Rechnungen. Da in diesem Fall nur zwei Stellen hinter dem Komma sinnvoll sind, kann man die Programmzeile zur Rundung ab dem Semikolon in

INT(100∗(Z+.005))/100

beziehungsweise

INT(100∗(VAL Z$+.005))/100

abändern.

Eine ebenso nützliche Funktion bei großen Rechnern ist der SORT-Befehl. Damit können Texte alphanumerisch sortiert werden. Auf dem ZX 81 wird dies mit einem elfzeiligen Programm erledigt. Die Variable X muß die Anzahl der zu sortierenden Wörter enthalten, und die Wörter selbst sind in dem Feld W$ gespeichert. Falls sich zwei identische Wörter unter den zu sortierenden befinden, muß folgende Zeile eingefügt werden:

55 GOTO 100.

```
 10 FOR I=1 TO X
 20 FOR J=1 TO X
 30 FOR K=1 TO LEN W$(I)
 40 IF W$(I,K)<>W$(J,K) THEN GOTO 60
 50 NEXT K
 60 IF W$(I,K)>W$(J,K) THEN GOTO 100
 70 LET K$=W$(I)
 80 LET W$(I)=W$(J)
 90 LET W$(J)=K$
100 NEXT J
110 NEXT I
```

Schnellere Eingabe für BASIC-Programme

Beim Texas Instruments TI99/4A muß man bei der Eingabe von BASIC-Programmen jeden Befehl Zeichen für Zeichen eintippen. Besitzt man den Modul TI-Extended BASIC, gibt es die Möglichkeit, über die Control-Taste in Verbindung mit einem bestimmten Zeichen einen BASIC-Befehl komplett einzugeben.

Die abgedruckte Tabelle zeigt, welcher Befehl über welche Control-Taste eingegeben werden kann. Einziger Nachteil bei dieser Methode ist, daß bei der Eingabe das Control-Zeichen nur durch ein Blank am Bildschirm angezeigt wird. Erst beim Ausgeben des Programmes über LIST werden die entsprechenden BASIC-Befehle angezeigt. Dieses sogenannte Short-Hand-BASIC funktioniert allerdings nur im Programmiermodus.

Tabelle

Taste	Basic-Anweisung
CTRL Z	REM
X	STOP
V	NEXT
CTRL B	(multiple statement Separator)
N	BREAK
M	LET
>	ON
A	ELSE
S	DATA
D	IF
F	GOTO
G	GOSUB
H	RETURN
K	END
L	FOR
;	PRINT
Q	UNTRACE
W	READ
E	GO
R	INPUT
T	RESTORE
Y	DELETE
U	RANDOMIZE
I	DEF
O	UNBREAK
P	TRACE
/	AND
1	TO
2	STEP
J	DIM
8	OPTION
9	OPEN
Ø	THEN
=	CALL
FCTN ;	NOT
/	OR
Ø	XOR

Scan: S. Höltgen

Für jeden etwas...

Wir haben uns vier Spiele ausgesucht und für Sie getestet

Verflixtes Schlangengewirr

„Hustle" übersetzt Langenscheidts Taschenwörterbuch mit „im Gedränge stoßen, drängen, treiben, sich drängen, eilen, mit Hochdruck arbeiten, sich dranhalten". Und ein Gedränge ist es wirklich, das bei diesem Spiel auf dem Bildschirm zu sehen ist. Ziel des Spieles ist es: „ein schlangenähnliches Objekt auf Ziele zu richten, aber ohne Ihren Gegenspieler, den Rand des Bildschirms oder sich selber zu treffen!" Rasche Reflexe und eine gute, vor allem aber schnelle Koordination zwischen Auge und Hand sind notwendig, um bei „Hustle" Punkte zu sammeln. Beim Spiel „Schlangenkampf" muß versucht werden, den „Punktezielen" auszuweichen und die gegnerische Schlange zu fressen... „Gefressene Punkteziele" werden dem Mitspieler zugerechnet. Je schneller der Spieler ein „Punktziel" erreicht hat, um so mehr Punkte können gesammelt werden. Je nachdem wie gespielt wird, erklingt eine Melodie, die sich zu einem wahren Konzert steigern kann. Für den Gewinner spielt der Computer einen Evergreen – den vom „Old Fellow". Das Spiel wird besonders interessant, wenn die „Blockade" angepeilt wird. Dann gilt es, die gegnerische Schlange in die Enge zu treiben. Stößt sie mit irgend etwas zusammen, erhält der Spieler 100 Zähler auf seinem Punktekonto gutgeschrieben.

Schwierigkeitsgrad: Vier verschiedene Varianten können gewählt werden. Von normal über Anfänger und Amateur bis hin zu „Profi".

Spieler: 1 oder 2

Unser Testurteil: Keine leichte Aufgabe, bei diesem Spiel Punkte zu sammeln. Jedenfalls hat uns das Spiel sehr viel Spaß gemacht. Wir meinen, die Herstellerangabe „ab 10 Jahre" ist korrekt. Aber auch Oldtimer unter den Spiele-Freaks werden noch ihre Freude daran haben. Vor allem die akustische Untermalung bringt Schwung in das Spiel. Wer schnell zu Schweißausbrüchen neigt, wird nach diesem Spiel schnell unter die Dusche gehen müssen. Mitdenken und Konzentration sind gefragt, um sich gegen die Schlange behaupten zu können. Wer einen Mitspieler aktivieren kann, wird mit „Hustle" vielleicht noch mehr Spaß als beim Kampf gegen den Computer haben. Fazit: Ein Spiel, das ohne kriegerische Handlungen spannend ist.

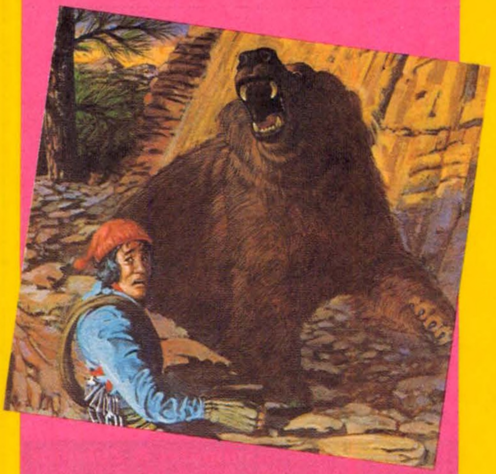
Hersteller: Texas Instruments

Geeignet für: TI 99/4A

Preis: 79 Mark

Die höchsten Gipfel erklimmen

Der Extrembergsteiger Reinhold Messner bekommt Konkurrenz. Allerdings nur auf dem Bildschirm. Der Held dieses Spiels ist nämlich ein ehrgeiziger Kletterer, der es sich zum Ziel gesetzt hat, die berühmten Gipfel dieser Welt zu erklimmen. Für Normalbergsteiger schon schwindelerrregend, beginnt es für unseren Helden noch fast wie ein Sonntagsspaziergang. Der Hood mit 3427 m Höhe steht an. Doch dann kommen ernsthafte alpine Schwierigkeiten: Matterhorn (4477 m), Kenya (5193 m), McKinley (6194 m), Garmo (7495 m) und schließlich der Mount Everest mit 8848 Höhenmetern. Unser Bergsteigerheld hat nun auch noch den Ehrgeiz, gegen die Uhr zu klettern. Deshalb werden Zeit und Höhenmeter einge-



blendet. Auch will er möglichst viele Höhenmeter auf sein Konto verbuchen. Das alles wäre ja noch nicht einmal so schlimm, wenn auf dem Weg nach oben nicht vielfältigste Gefahren lauern würden. Da gibt es Steinschlag, Eisfelder, Lawinen und gar noch Bären, die immer wieder des Weges kreuzen. Wie gut, daß eine Art Schneekönig Yeti auf allen Bergen wohnen muß. Mit einer synthetischen Stimme warnt er vor Gefahren. Zum Beispiel vor Berglöwen, Bären und Fledermäusen, die den wackeren Alpinisten abstürzen lassen können. Schön, daß

Scan: S...

Spiele Test

er nicht zerschmettert unten liegen bleibt, sondern so weich fällt, daß er den Aufstieg wiederholen kann.

Schwierigkeitsgrad: Achtzehn Schwierigkeitsstufen werden angeboten.

Spieler: 1 oder 2

Unser Testurteil: Ein Spiel, bei dem die Spannung einmal nicht in das Abknallen von außerirdischen Eindringlingen eingebaut ist. Bergsteigen, sonst ein eher „langweiliger" Sport, wird sehr spannend dargeboten. Damit der Spaß erhalten bleibt, sind in das Spiel Gags eingebaut, bis hin zum Gipfelbild mit dem „Erstbesteiger", der die berühmte Fahne in der Hand hält. Echte Alpinisten werden sicher bei den eingebauten Gefahren nur schmunzeln. Da ist die Phantasie des Programmierers etwas durchgegangen.

Hersteller: TI

Geeignet für: TI 99/4A

Preis: 99 Mark

Fast wie Formel I fahren

Atari hat wieder einmal einen Standard gesetzt – für Autorennen auf dem Bildschirm. Mit „Pole Position" wird ein ziemlich realistisches Autorennen auf einer anspruchsvollen Rennstrecke simuliert. Der rennbegeisterte Computerspieler kann mit Hilfe des Joystick die Geschwindigkeit bestimmen, schalten, überholen, sich in der Technik des Kurvenfahrens üben und sogar ausbremsen. Super die Rennstrecke, die wirklich Konzentration verlangt. Und dann die Geschwindigkeit, mit der über die Piste gedonnert wird. Da huschen die Seitenbegrenzungen nur noch schemenhaft vorbei. Um sich für einen Startplatz im Grand Prix zu qualifizieren, müssen erst die Qualifikationen bestanden werden. Das heißt, die schwierige Rennstrecke in 73 Sekunden oder weniger zu durchfahren. Schon hier wird die Spreu vom Weizen getrennt. Und nur Superfahrer am Bildschirm sichern sich einen Startplatz. Auf dem Bildschirm werden der gewählte Gang, der Punktestandzähler, der Runden-Zeitmesser, der Tachometer und der Wettrenn-Zeitmesser eingeblendet. Da weiß der Computer-Fahrer ständig, wie es um seine Position und vor allem um sein Fahrvermögen bestellt ist. Pole Position macht auch jedem Spieler schnell deutlich, daß noch kein Meister vom Himmel gefallen ist. Wer sich dem Ende einer Runde nähert, erkennt das durch blinkende karierte Flaggen, wie im echten Rennen.

Schwierigkeitsgrad. Zwei Gänge bestimmen zwei Geschwindigkeiten.

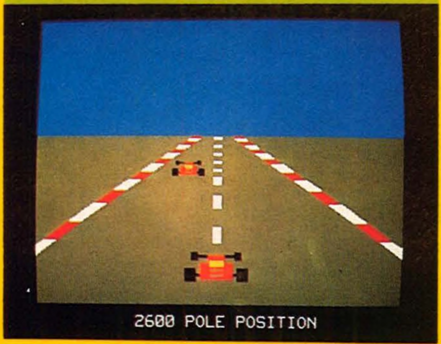
Spieler: 1

Unser Testurteil: Um das Resümee vorwegzunehmen: ein Superspiel. Pole Position gehört nicht zu jenen Spielen, die nach ein paar Tagen bereits langweilig werden und die in ihrem System schnell zu durchschauen sind. Im Gegenteil. Gefallen hat uns die gute farbliche Darstellung des Renngeschehens und die Möglichkeit, Geschwindigkeiten selbst zu bestimmen. Selten haben wir eine Kassette so lange getestet. Oder, um es ehrlich zu sagen: Das Pole-Position-Fieber hat uns erfaßt und nicht mehr losgelassen. Wahrscheinlich findet sich schwer eine Medizin gegen diese „Krankheit".

Hersteller: Atari Electronic

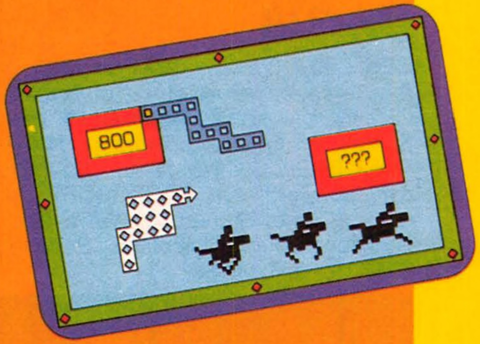
Geeignet für: Atari 400/800, 600 XL

Preis: 149 Mark



Jagd in der Höhle

Das Spiel ist zu einer aufregenden Jagd in einem verborgenen Labyrinth von Höhlen und verschlungenen Wegen gestaltet. Auf dem Weg zur Wumpus-Stadt lauert eine Menge Gefahren in dem Labyrinth aus kleinen Höhlen. Es gibt einige Richtlinien, um sich zurechtzufinden. Wer in die Nähe von Wumpus kommt, bemerkt dies an Blutspuren in der Höhle. Und vor der Wumpus-Höhle sind im-



mer zwei Blutspuren zu erkennen. Zum Schluß gilt es, den Wumpus genau auszumachen. Wer glaubt, ihn geortet zu haben, der muß einen Pfeil auf das Versteck des Wumpus abschießen. Doch aufgepaßt. Ein Fehlschuß, und das Spiel gilt als verloren. Jeder Spieler hat nämlich nur einen Pfeil zur Verfügung. Ja, wer hätte das gedacht, daß die Wumpus-Jagd so schwer sein kann. Kaninchen in freier Wildbahn scheinen dem Jäger leichter vor die Flinte zu laufen.

Schwierigkeitsgrad: leicht, mittel und schwer.

Spieler: 1

Unser Testurteil: Das Spiel ist bisher in die Sprachen Niederländisch, Englisch, Französisch, Deutsch, Italienisch und Schwedisch übersetzt worden. Ein Zeichen dafür, daß sich der Hersteller höhere Absatzzahlen verspricht. Das kann er auch. Und er muß auch nicht bangen, daß ihm das Spiel als „kriegerisch" um die Ohren gehaut wird. TI sollte nicht bei der Bedienungsanleitung knausern. Die ist sehr dürftig und sollte verbessert werden.

Hersteller: Texas Instruments

Geeignet für: TI 99/4A

Preis: 59 Mark

Scan: S. Höltgen

Vokabeltrainer

Ein Vokabellernprogramm für den TI-99/4A mit Extended BASIC. Neben Standardfunktionen enthält es noch ein didaktisch besonders interessantes Lernspiel

Nach dem Programmstart erscheint das Hauptmenü.

Vokabeleingabe

Es werden sämtliche bereits gespeicherten Vokabeln (falls überhaupt schon welche vorhanden sind) aus dem Speicher gelöscht und neue Vokabeln (maximal 100) in den Speicher aufgenommen. Dabei wird bei jeder Vokabel vom Programm zuerst der deutsche, dann der fremdsprachige Teil erfragt. Drückt man beim deutschen Teil der Vokabel ENTER ohne Eingabe, so betrachtet das Programm die Eingabe als abgeschlossen.

Will man eine Vokabel eingeben, die wortgleich bereits als fremdsprachige oder deutsche Vokabel gespeichert ist, so wird sie ignoriert. Dies hat den Zweck, Mehrdeutigkeiten bei der Vokabelabfrage oder beim Lernspiel oder anderen Funktionen auszuschließen.

Ebenfalls ignoriert wird eine Vokabel, falls der fremdsprachige Teil nur aus dem Nullstring (ENTER ohne Eingabe) besteht, da ENTER ohne Eingabe bei anderen Funktionen eine Spezialaufgabe erfüllt.

Weitere Vokabeln

Es werden zu den bereits gespeicherten noch weitere mitaufgenommen. Also alte Vokabeln werden nicht gelöscht. Ansonsten gilt das gleiche, wie für die Vokabeleingabe.

Ansehen

Nacheinander werden alle gespeicherten Vokabeln jeweils für kurze Zeit angezeigt.

```
100 !VOKABELTRAINER
110 !(C) 14.10.83 BY
120 !FRANK SCHWELLINGER
130 !NEUWIESENSTR. 7
140 !7535 STEIN
150 CALL CHAR(112,"49EA6E38783EF706")
160 CALL COLOR(11,9,1)
170 RANDOMIZE
180 CALL SCREEN(16)
190 CALL MAGNIFY(3)
200 CALL CHAR(100,"0000000000003030000000000000000000000000004000003030000000000000
000")
210 CALL CHAR(104,"000F1931FFFF"&RPT$("0",22)&"C02010FFFF"&RPT$("0",20))
220 CALL CHAR(96,RPT$("F",16)):: CALL COLOR(9,11,11)
230 LK=3
240 OPTION BASE 1
250 DIM V$(100,2),ST(100)
260 CALL CLEAR
270 DISPLAY AT(1,1):"DRUECKEN SIE"
280 DISPLAY AT(5,1):"1       VOKABELEINGABE"
290 DISPLAY AT(7,1):"2       WEITERE VOKABELN"
300 DISPLAY AT(9,1):"3       VOKABELN ANSEHEN"
310 DISPLAY AT(11,1):"4       VOKABEL LOESCHEN"
320 DISPLAY AT(12,3):"FUER"
330 DISPLAY AT(13,1):"5       VOKABEL KORRIGIEREN"
340 DISPLAY AT(15,1):"6       VOKABEL UEBERSETZEN"
350 DISPLAY AT(17,1):"7       VOKABELLERNSPIEL"
360 DISPLAY AT(19,1):"8       VOKABELN ABFRAGEN"
370 CALL KEY(0,K,S):: IF S=0 THEN 370
380 IF ABS(K-52.5)>4 THEN CALL SOUND(200,200,3):: GOTO 370
390 ON K-48 GOSUB 410,440,560,640,830,970,1130,2430
400 GOTO 260
410 ! VOKABELEINGABE
420 CALL CLEAR
430 VZ=0
440 !VOKABELN HINZUFUEGEN
450 VZ=VZ+1
460 IF VZ=101 THEN DISPLAY AT(14,1):"ES KOENNEN NUN KEINE WEITE- REN VOKABELN ME
HR EINGEGEBENWERDEN!" :: CALL PAUSE(12):: VZ=100 :: RETURN
470 DISPLAY AT(3,1)ERASE ALL:"DEUTSCHE VOKABEL"
480 ACCEPT AT(6,1)VALIDATE(UALPHA,DIGIT,"-")SIZE(27):V$(VZ,1)
490 IF V$(VZ,1)="" THEN VZ=VZ-1 :: CALL CLEAR :: DISPLAY AT(14,1):"EINGABE ABGES
CHLOSSEN" :: CALL PAUSE(4):: RETURN
500 T$=V$(VZ,1):: GOTO 2600
510 DISPLAY AT(12,1):"ENGLISCHE VOKABEL"
520 ACCEPT AT(15,1)VALIDATE(UALPHA,DIGIT,"-")SIZE(27):V$(VZ,2)
530 T$=V$(VZ,2):: GOTO 2660
540 ST(VZ)=1
550 GOTO 450
560 !VOKABELN ANSEHEN
570 FOR I=1 TO VZ
580 CALL CLEAR
590 DISPLAY AT(11,INT((29-LEN(V$(I,1)))/2)):V$(I,1)
600 DISPLAY AT(13,14):"="
610 DISPLAY AT(15,INT((29-LEN(V$(I,2)))/2)):V$(I,2):: CALL PAUSE(2):: CALL KEY(0
,Q,W):: IF Q=13 THEN CALL CLEAR :: RETURN ELSE IF W THEN 610
620 NEXT I
630 RETURN
640 !VOKABEL LOESCHEN
650 IF VZ=0 THEN CALL CLEAR :: RETURN
660 DISPLAY AT(10,1)ERASE ALL:"WELCHE VOKABEL SOLL GE-      LOESCHT WERDEN?"
670 ACCEPT AT(15,1):W$
680 CALL CLEAR
690 FOR I=1 TO VZ
700 IF V$(I,1)=W$ OR V$(I,2)=W$ THEN 750
710 NEXT I
720 DISPLAY AT(12,1)ERASE ALL:"DIESE VOKABEL IST NICHT VOR-HANDEN"
730 CALL PAUSE(1)
740 RETURN
750 GEF=I
```

Scan: S. Höltgen


```
760 FOR I=GEF TO VZ-1
770 V$(I,1)=V$(I+1,1)
780 V$(I,2)=V$(I+1,2)
790 ST(I)=ST(I+1)
800 NEXT I
810 VZ=VZ-1
820 RETURN
830 !VOKABEL AENDERN
840 IF VZ=0 THEN CALL CLEAR :: RETURN
850 DISPLAY AT(10,1)ERASE ALL:"WELCHE VOKABEL SOLL GEANDERTWERDEN?"
860 ACCEPT AT(13,1):W$
870 FOR J=1 TO 2
880 FOR I=1 TO VZ
890 IF V$(I,J)=W$ THEN 940
900 NEXT I
910 NEXT J
920 DISPLAY AT(12,1)ERASE ALL:"DIESE VOKABEL IST NICHT VOR-HANDEN" :: CALL PAUSE
(2)
930 RETURN
940 DISPLAY AT(10,1)ERASE ALL:"WIE HEISST DIE GEANDERTE VO-KABEL?"
950 ACCEPT AT(13,1)VALIDATE(UALPHA,DIGIT,"-"):V$(I,J)
960 RETURN
970 !VOKABEL UEBERSETZEN
980 IF VZ=0 THEN CALL CLEAR :: RETURN
990 DISPLAY AT(7,1)ERASE ALL:"WELCHE VOKABEL SOLL UEBER-  SETZT WERDEN?"
1000 ACCEPT AT(10,1):W$
1010 FOR J=1 TO 2
1020 FOR I=1 TO VZ
1030 IF V$(I,J)=W$ THEN 1080
1040 NEXT I
1050 NEXT J
1060 DISPLAY AT(12,1)ERASE ALL:"VOKABEL NICHT VORHANDEN" :: CALL PAUSE(2)
1070 RETURN
1080 DISPLAY AT(9,INT((29-LEN(W$))/2))ERASE ALL:W$
1090 DISPLAY AT(13,14):"="
1100 DISPLAY AT(17,INT((29-LEN(V$(I,ABS(J-2)+1)))/2)):V$(I,ABS(J-2)+1)
1110 CALL PAUSE(4)
1120 RETURN
1130 !HANGMAN
1140 CALL CLEAR
1150 IF VZ=0 THEN RETURN
1160 FOR I=1 TO VZ
1170 ST(I)=0
1180 NEXT I
1190 OG=VZ
1200 DISPLAY AT(4,1):"WAEHLEN SIE DIE TRAININGS  -SPIELART"
1210 DISPLAY AT(7,1):"DRUECKEN SIE"
1220 DISPLAY AT(10,1):"1      ENGLISCH-DEUTSCH"
1230 DISPLAY AT(12,1):"2      DEUTSCH-ENGLISCH"
1240 DISPLAY AT(13,3):"FUER"
1250 DISPLAY AT(14,1):"3      GEMISCHT"
1260 DISPLAY AT(16,1):"4      REINES SPIEL"
1270 CALL KEY(0,K,S):: IF S=0 THEN 1270
1280 IF ABS(K-50.5)>1.5 THEN CALL SOUND(200,220,4):: GOTO 1270
1290 ON K-48 GOSUB 1310,1610,1620,1630
1300 RETURN
1310 S1,S2=0
1320 CALL CLEAR
1330 CALL HCHAR(2,1,96,30)
1340 CALL SPRITE(#2,104,9,2,16)
1350 CALL SPRITE(#1,100,2,2,16)
1360 POSI=16
1370 P=INT(OG*RND+1)
1380 IF S2 THEN S1=INT(2*RND)
1390 DISPLAY AT(17,1)SIZE(28):""
1400 DISPLAY AT(10,1)SIZE(28):""
1410 DISPLAY AT(17-7*S1,INT((29-LEN(V$(P,2-S1)))/2)):V$(P,2-S1)
1420 DISPLAY AT(10+7*S1,INT((29-LEN(V$(P,1+S1)))/2)):RPT$("_",LEN(V$(P,1+S1)))
1430 R$=V$(P,1+S1)
1440 GUT=-1
1450 CALL KEY(0,K,S):: IF S<>1 THEN 1450
1460 IF K=13 THEN CALL DELSPRITE(ALL):: CALL CLEAR :: RETURN
1470 IF 0=POS(R$,CHR$(K),1)THEN GOSUB 1940
1480 IF POSI=232 THEN RETURN
1490 IF POS(R$,CHR$(K),1)THEN GOSUB 1600 :: R$=SEG$(R$,1,POS(R$,CHR$(K),1)-1)&"!
"&SEG$(R$,1+POS(R$,CHR$(K),1),28):: GOTO 1490
1500 IF R$<>RPT$("!",LEN(R$))THEN 1450
1510 IF GUT THEN ST(P)=ST(P)+1
1520 IF ST(P)>=LK THEN GOSUB 2030
1530 IF OG>0 THEN 1370
1540 DISPLAY AT(10,1)SIZE(28):""
1550 DISPLAY AT(17,1)SIZE(28):""
1560 DISPLAY AT(12,1):"GUT! DU SCHEINST JETZT ALLE VOKABELN ZU KOENNEN. DEIN   A
UTO STEHT ";INT((232-POSI)/3);" M VOR DEM": :"ABGRUND"
1570 CALL PAUSE(6)
1580 CALL DELSPRITE(ALL)
1590 RETURN
1600 DISPLAY AT(10+7*S1,INT((29-LEN(V$(P,1+S1)))/2)-1+POS(R$,CHR$(K),1))SIZE(1):
CHR$(K):: CALL SOUND(100,660,2):: RETURN
1610 S1=1 :: S2=0 :: GOTO 1320
```

Drückt man während des Vokabelansehens für einige Sekunden die Taste ENTER, dann wird das Vokabelansehen abgebrochen und ins Hauptmenü zurückgekehrt. Drückt man eine beliebige andere Taste, so kann man sich die augenblicklich angezeigte Vokabel so lange anschauen, wie man die Taste gedrückt hält.

Löschen

Man kann einzelne Vokabeln aus dem Speicher entfernen. Das Programm erfragt dann die zu löschende Vokabel. Nun hat man freie Wahl, ob man die fremdsprachige oder die deutsche Vokabel eingeben will, es werden natürlich in jedem Fall beide gelöscht. War die zu löschende Vokabel nicht gespeichert, so wird man darauf hingewiesen.

Korrigieren

Stellt man fest, daß ein Fehler in einer Vokabel vorhanden ist, so läßt sich die Vokabel noch nachträglich korrigieren. Dazu muß zuerst die falsch geschriebene Vokabel und dann die neue eingegeben werden.

Übersetzen

Das Programm fragt nach der zu übersetzenden Vokabel, wobei es einem freisteht, eine fremdsprachige oder deutsche Vokabel einzugeben.

Abfragen

Es erscheint ein weiteres Menü. Mit ihm kann gewählt werden, ob das Programm nur nach fremdsprachigen, nur nach englischen Vokabeln oder gemischt abfragen soll. Daraufhin gibt das Programm Vokabeln vor, die man übersetzen soll. Wenn das Programm den Eindruck hat, das man alle Vokabeln kann, dann hört es auf mit dem Vokabelabfragen. Nach einer falschen Antwort, zeigt einem das Programm die richtige Antwort.

Will man die Vokabelabfrage vorzeitig abbrechen, so gibt man einfach keine Vokabel mehr ein, sondern drückt EN-

Scan: S. Höltgen

TER ohne Eingabe. Das Vokabelabfragen ist gedacht für den Fall, daß man die Vokabeln schon einigermaßen sicher beherrscht und nun noch schnell nachprüfen will, ob man tatsächlich fit ist oder ob man noch ein wenig lernen sollte.

Vokabellernspiel

Es handelt sich hier um den Lernschritt, der zwischen dem Anschauen der Vokabeln und deren perfekter Beherrschung liegt. Im folgenden wird erklärt, wie das Spiel gehandhabt wird und worin sein großer Vorteil gegenüber anderen Vokabellern-Methoden liegt.

Bei der Wahl des Vokabellernspiels erscheint zuerst ein Untermenü. Dieses Untermenü läßt insgesamt vier Auswahlmöglichkeiten zu. Die drei ersten Möglichkeiten sind ähnlich wie die beim Vokabelabfragen. Der Computer gibt je nach Wahl fremdsprachige oder deutsche oder gemischt Vokabeln vor, die zu übersetzen sind. Allerdings zeigt der Computer auf dem Bildschirm für jeden Buchstaben und jedes Leerzeichen, das die zu erratende Vokabel hat, einen Strich an, wie bei dem Spiel „Hangman". Und genauso wie bei diesem Spiel muß die Vokabel auch buchstabenweise erraten werden.

Das Spiel ist erst beendet, wenn das Programm den Eindruck hat, daß man alle Vokabeln kann, oder wenn man einen tragischen Unfall erleidet. Am oberen Bildschirmrand befindet sich nämlich zu Spielbeginn ein Auto, dessen Insasse man ist, 72 m vor einem tiefen Abgrund. Für jeden falschen Buchstaben, den man eingibt, fährt das Auto einen Meter näher auf den Abgrund zu.

Die vierte Spielmöglichkeit besteht nun darin, daß wieder oben ein Auto steht, daß aber auf dem Bildschirm zwei Strichreihen erscheinen. Die obere Reihe repräsentiert das deutsche Wort, die untere das fremdsprachige. Obwohl es sich hierbei um ein reines Spiel handelt, muß man sich auch hierbei intensiv mit den Vokabeln beschäftigen.

```
1620 S2=1 :: GOTO 1320
1630 CALL CLEAR
1640 CALL HCHAR(2,1,96,30)
1650 CALL SPRITE(#2,104,9,2,16)
1660 CALL SPRITE(#1,100,2,2,16)
1670 POSI=16
1680 FOR WE=1 TO 18
1690 P=INT(VZ*RND+1)
1700 DISPLAY AT(17,1)SIZE(28):""
1710 DISPLAY AT(10,1)SIZE(28):""
1720 DISPLAY AT(17,INT((29-LEN(V$(P,2)))/2)):RPT$("_",LEN(V$(P,2)))
1730 DISPLAY AT(10,INT((29-LEN(V$(P,1)))/2)):RPT$("_",LEN(V$(P,1)))
1740 R1$=V$(P,1)
1750 R2$=V$(P,2)
1760 CALL KEY(0,K,S):: IF S<>1 THEN 1760
1770 IF K=13 THEN CALL DELSPRITE(ALL):: CALL CLEAR :: RETURN
1780 IF 0=POS(R1$,CHR$(K),1)AND 0=POS(R2$,CHR$(K),1)THEN GOSUB 1940
1790 IF POSI=232 THEN RETURN
1800 S1=0
1810 R$=R1$
1820 IF POS(R$,CHR$(K),1)THEN GOSUB 1600 :: R$=SEG$(R$,1,POS(R$,CHR$(K),1)-1)&"!
"&SEG$(R$,1+POS(R$,CHR$(K),1),28):: GOTO 1820
1830 R1$=R$
1840 S1=1
1850 R$=R2$
1860 IF POS(R$,CHR$(K),1)THEN GOSUB 1600 :: R$=SEG$(R$,1,POS(R$,CHR$(K),1)-1)&"!
"&SEG$(R$,1+POS(R$,CHR$(K),1),28):: GOTO 1860
1870 R2$=R$
1880 IF R1$<>RPT$("!",LEN(R1$))OR R2$<>RPT$("!",LEN(R2$))THEN 1760
1890 NEXT WE
1900 DISPLAY AT(10,1)SIZE(28):""
1910 DISPLAY AT(17,1)SIZE(28):""
1920 DISPLAY AT(12,1):"DU STEHST ";INT((232-POSI)/3);" M VOR DEM": :"ABGRUND"
1930 GOTO 1570
1940 GUT=0 :: ST(P)=0 :: POSI=POSI+3
1950 IF POSI=232 THEN 1980
1960 CALL LOCATE(#2,2,POSI,#1,2,POSI)
1970 RETURN
1980 CALL MOTION(#1,20,0,#2,20,0)
1990 FOR I=1 TO 50
2000 CALL SOUND(-1,1000-10*I,0)
2010 NEXT I
2020 CALL SOUND(1000,-7,0):: CALL DELSPRITE(ALL):: CALL HCHAR(24,30,112,1):: GOS
UB 2610 :: RETURN
2030 OG=OG-1
2040 IF OG<1 THEN RETURN
2050 FOR DC=VZ TO P+1 STEP -1
2060 IF ST(DC)<3 THEN 2090
2070 NEXT DC
2080 RETURN
2090 HI=ST(P)
2100 ST(P)=ST(DC)
2110 ST(DC)=HI
2120 HI$=V$(P,1)
2130 V$(P,1)=V$(DC,1)
2140 V$(DC,1)=HI$
2150 HI$=V$(P,2)
2160 V$(P,2)=V$(DC,2)
2170 V$(DC,2)=HI$
2180 RETURN
2190 S1,S2=0
2200 CALL CLEAR
2210 FOR II=1 TO LK
2220 COUNTER=VZ
2230 FOR J=1 TO VZ
2240 ST(J)=0
2250 NEXT J
2260 IF S2 THEN S1=INT(2*RND)
2270 P=INT(VZ*RND+1)
2280 IF ST(P)=1 THEN 2270
2290 CALL CLEAR
2300 DISPLAY AT(10,INT((29-LEN(V$(P,2-S1)))/2)):V$(P,2-S1)
2310 ACCEPT AT(17,1):E$
2320 IF E$="" THEN CALL CLEAR :: RETURN
2330 ST(P)=1 :: COUNTER=COUNTER-1
2340 IF E$<>V$(P,1+S1)THEN GOSUB 2540 ELSE DISPLAY AT(17,10)ERASE ALL:"RICHTIG!"
2350 CALL PAUSE(2)
2360 IF COUNTER>0 THEN 2260
2370 NEXT II
2380 RETURN
2390 S1=1 :: S2=0
2400 GOTO 2200
2410 S2=1
2420 GOTO 2200
2430 !VOKABELABFRAGEN
2440 CALL CLEAR
2450 IF VZ=0 THEN RETURN
2460 DISPLAY AT(2,1):"DRUECKEN SIE BITTE"
2470 DISPLAY AT(10,1):"1       ENGLISCH-DEUTSCH"
2480 DISPLAY AT(15,1):"2 FUER DEUTSCH-ENGLISCH"
```

Scan: S. Höltgen


```
2490 DISPLAY AT(20,1):"3       GEMISCHT"
2500 CALL KEY(0,K,S):: IF S=0 THEN 2500
2510 IF ABS(K-50)>1 THEN CALL SOUND(200,200,3):: GOTO 2500
2520 ON K-48 GOSUB 2190,2390,2410
2530 RETURN
2540 ST(P)=0
2550 COUNTER=COUNTER+1
2560 DISPLAY AT(7,7)ERASE ALL:"LEIDER FALSCH!"
2570 DISPLAY AT(10,1):"RICHTIG WAERE GEWESEN"
2580 DISPLAY AT(20,INT((29-LEN(V$(P,1+S1)))/2)):V$(P,1+S1)
2590 RETURN
2600 FOR I=1 TO VZ-1 :: IF T$=V$(I,1)OR T$=V$(I,2)THEN CALL SOUND(500,110,0):: V
Z=VZ-1 :: GOTO 450 ELSE 2650
2610 FOR IN=1 TO 3 :: CALL SOUND(750,110,0):: CALL PAUSE(.5):: NEXT IN
2620 CALL SOUND(500,131,0):: CALL SOUND(250,123,0):: CALL SOUND(500,123,0):: CAL
L SOUND(500,110,0):: CALL SOUND(250,110,0)
2630 CALL SOUND(250,110,0)
2640 RETURN
2650 NEXT I :: GOTO 510
2660 IF T$="" THEN CALL SOUND(500,110,0):: VZ=VZ-1 :: GOTO 450
2670 FOR I=1 TO VZ-1 :: IF T$=V$(I,1)OR T$=V$(I,2)THEN CALL SOUND(500,110,0):: V
Z=VZ-1 :: GOTO 450
2680 NEXT I :: GOTO 540
2690 SUB PAUSE(S)
2700 FOR I=1 TO 278*S :: NEXT I
2710 SUBEND
```

Jede Variante des Spiels läßt sich jederzeit abbrechen, indem man anstatt eine Buchstabentaste zu drücken die Taste ENTER drückt.

Wahl der Reihenfolge zum Lernen

Die beste Reihenfolge, um Vokabeln zu lernen, besteht in

1. V. ansehen
2. Lernspiel (1–3)
3. Lernspiel 4
4. Abfragen
5. Lernspiel 4

Und nun viel Spaß beim Vokabellernen!

Frank Schwellinger

Invaders 64

Ein für den Commodore 64 mit Joystick geschriebenes Actionspiel. Es ist ganz in Maschinensprache geschrieben und verblüfft durch seinen professionellen Spielhallen-Charakter

Das Maschinenprogramm befindet sich verschlüsselt in Dezimalzahlen innerhalb von DATA-Zeilen. Weil wegen seines großen Umfangs sich leicht Tippfehler einschleichen können, wird in einer Einlese-Routine zu Beginn des Programms sowohl die Anzahl der eingegebenen Bytes kontrolliert als auch eine Prüfsumme gebildet.

```
READY.

1000 REM INVADERS 64
1010 REM 10.1983 BY PETER MENKE
1020 :
1030 S=4*4096+7*256:PRINT"□"
1040 READX:IF X=-1THEN1060
1050 A=A+X:POKES+B,X:B=B+1:GOTO1040
1060 IF B>2030 THENPRINT"**ZUVIELE DATEN**":END
1070 IF B<2030 THENPRINT"**ZUWENIG DATEN**":END
1080 IFA<>210762THENPRINT"**DATEN ANZAHL RICHTIG**"
     :GOTO1130
1090 PRINT"**ALLES RICHTIG **
1100 PRINT"**ZUM START SPACE**"
1110 GET A$:IFA$<>" "THEN1110
1120 SYS S
1130 PRINT"**JEDOCH EINIGE FALSCH**":END
1140 :
1150 DATA 169,0,133,254,169
1160 DATA 56,133,255,169,0
1170 DATA 162,13,160,0,145
1180 DATA 254,136,208,251,230
1190 DATA 255,202,208,244,169
1200 DATA 0,133,254,169,216
1210 DATA 133,255,169,6,162
1220 DATA 4,160,0,145,254
1230 DATA 136,208,251,230,255
1240 DATA 202,208,244,160,63
1250 DATA 185,73,71,153,0
1260 DATA 56,136,16,247,173
1270 DATA 24,208,41,240,24
1280 DATA 105,14,141,24,208
1290 DATA 76,137,71,126,153
```

```
1300 DATA 153,255,255,153,153
1310 DATA 129,102,165,36,126
1320 DATA 153,153,255,126,204
1330 DATA 51,204,51,204,51
1340 DATA 204,51,0,0,130
1350 DATA 42,204,51,204,51
1360 DATA 0,0,0,0,0
1370 DATA 145,205,51,0,24
1380 DATA 24,24,24,24,24
1390 DATA 24,16,8,16,8
1400 DATA 16,8,16,8,24
1410 DATA 60,126,24,60,126
1420 DATA 255,102,162,7,189
1430 DATA 151,71,157,184,56
1440 DATA 202,16,247,76,159
1450 DATA 71,66,66,66,255
1460 DATA 66,66,66,255,162
1470 DATA 161,189,71,78,157
1480 DATA 127,57,202,208,247
1490 DATA 76,221,71,141,24
1500 DATA 212,162,5,138,24
1510 DATA 105,58,157,0,4
1520 DATA 157,27,4,202,16
1530 DATA 243,162,3,138,24
1540 DATA 105,64,157,23,4
1550 DATA 202,16,246,96,255
1560 DATA 255,255,255,255,255
1570 DATA 255,255,255,255,255
1580 DATA 2,255,255,255,255
1590 DATA 255,234,32,43,76
1600 DATA 169,0,141,12,64
1610 DATA 141,13,64,141,14
1620 DATA 64,32,222,76,32
1630 DATA 2,76,32,47,72
1640 DATA 32,73,72,32,150
1650 DATA 72,32,240,77,32
1660 DATA 221,72,32,97,76
1670 DATA 169,0,32,13,76
1680 DATA 32,62,73,32,174
1690 DATA 74,32,105,74,32
1700 DATA 5,75,32,34,75
```

Scan: S. Höltgen

Für jeden etwas...

Wir haben uns vier Spiele ausgesucht und für Sie getestet

Hilfe, die Galaxier kommen

Wer bei diesem Spiel an Asterix und Obelix denkt, ist völlig schief gewickelt. Der Witz, der bei den Kämpfen der beiden Gallier gegen die Römer im Spiel ist, hat hier keinen Platz. Galaxian ist ein typisches Weltraumspiel. Es gilt, den Angreifern aus dem Weltraum wirkungsvoll entgegenzutreten. Die Galaxier ziehen, nach Rang und Namen geordnet, in voller Kriegsformation auf. Und da gibt es Militärränge, die in keinem Lexikon stehen. Außerirdische haben eben auf der Mutter Erde noch nicht die Kriegsschauplätze bevölkert und dadurch auch keinen Stoff für Lexikonschreiber und Historiker geliefert. Deshalb hier die Kämpfer beziehungsweise ihre Waffen: Drohnen, Geheimboten, Geleitschutz und Flaggschiffe. Um möglichst viele „Bonuspunkte" zu sammeln, ist es wichtig, zuerst den doppelten Geleitschutz und dann das Flaggschiff zu vernichten. Gibt 800 Punkte. Zum Vergleich: Der Abschuß eines Flaggschiffes mit Geleitschutz bringt nur 200 Punkte. Aber wie wird nun abgeschossen? Der Spieler beginnt mit drei Raumschiffen. Mit dem Steuerknüppel wird das Raumschiff in waagrechter Richtung nach rechts oder links auf dem Bildschirm bewegt. Mit dem Auslöseknopf des Steuerknüppels können Flugkörper zur Abwehr der Galaxier abgefeuert werden.

Schwierigkeitsgrad: Neun verschiedene Spielvarianten können gewählt werden.

Spieler: 1 oder 2

Unser Testurteil: Galaxian ist sicher nicht mehr der letzte Schrei auf dem Spiele-Markt, insbesondere auf dem Markt der Weltraumspiele. Aber es gehört ohne Zweifel zu den bewährtesten und beliebtesten. Und es hat viele Nachahmer gefunden. Meist wurde das Spielmuster mit den wellenartig angreifenden Außerirdischen übernommen. Die einzelnen Spielfiguren sind als Pictogramme, also in der bekannten Form von Computer-Männchen, dargestellt. Neuere Computerspiele arbeiten bereits mit Figuren, die den wirklichen Umrissen sehr ähnlich sind. Ein Flugzeug ist dann eben als Flugzeug zu erkennen. Bei Galaxian fühlt man sich etwas in die Gründerzeit der Computerspiele zurückversetzt.

Hersteller: Atari

Geeignet für: Atari 400 oder 800

Preis: 149 Mark

Grand Prix auf dem Bildschirm

Der Traum jedes Jungen, jedes Mannes – schnelle Autos. Am besten gleich jene Flitzer, die bei Grand-Prix-Rennen über die Pisten sausen. In Miniausführung und im Wohnzimmer kann dieser Traum wahr werden. Auf dem Bildschirm mit „Car Wars" von Texas Instruments. Konzentrations- und Reaktionsfähigkeit lassen sich mit diesem Spiel bei einem Autorennen gegen den Computer trainieren. Während des Rennens verdoppelt das Computerauto die Geschwindigkeit. Es ist nicht leicht, Zusammenstöße zu vermeiden. Ein Lenkfehler, und der Zusammenstoß auf der labyrinthähnlichen Piste kann nicht mehr vermieden werden. Dann spielt sich alles wie beim echten Rennen ab. Die hohe Geschwindigkeit zerfetzt die beiden Wagen, und zurück bleiben Trümmerteile und ein riesiger Ölfleck. Car Wars ist ein Spiel, bei dem es gilt, einen sehr gewitzten Konkurrenten auszumanövrieren. Dabei ist die Darstellung auf dem Bildschirm mit einer guten Lautuntermalung verstärkt.

Schwierigkeitsgrad: 3 stehen zur Auswahl (langsame Fahrt, schnelle Fahrt und extrem schnelle Fahrt)

Spieler: 1

Unser Testurteil: Autorennen gehören zur „Standardausrüstung" jeder Computerspiele-Sammlung. So nimmt auch fast jeder Hersteller – ob Videospiel oder Home-Computer – ein „Autorennen" in sein Spiele-Programm auf. Deshalb lassen sich hier besonders gut Vergleiche ziehen. Nach unserer Meinung gehört das hier getestete nicht zu den „raffiniertesten", wie sie jetzt immer mehr auf den Markt kommen. Sicher, die realistische Darstellung verdient Punkte. Aber die Konkurrenz hat nicht geschlafen. Gefallen hat uns der „Sound", der auch auf Rennpisten nicht besser eingefangen werden könnte. Ein Spiel, an dem man seine Freude haben kann. Vor allem wegen der eindrucksvollen Geräuschkulisse.

Hersteller: Texas Instruments

Geeignet für: TI 99/4A

Preis: 99 Mark

Scan: S. Höltgen

Gleichgesinnte unter sich

Austausch von Erfahrungen, Fachwissen oder Programmen – Clubs bieten die Möglichkeit, mehr über einen Computer zu erfahren. Hier ein Porträt vom TI99-Club-Journal

Nachdem immer deutlicher wurde, daß fast alle TI99/4A-Besitzer isoliert ihrem Hobby nachgehen und mancher sogar durch Inserate in Fachmagazinen Kontakt zu Gleichgesinnten suchen mußte, entstand die Idee zur Gründung eines umfassenden Clubs.

Erfahrungsaustausch

Doch wie soll man immerhin Tausende von TI-Benutzer erreichen und verbinden, zumal durch die breite Streuung ein Clubbetrieb im herkömmlichen Sinn kaum in Frage kommt. So wurde im Januar 1983 das TI99-Journal als Forum für Problemanalysen, Erfahrungsaustausch, Software-Vermittlung und beratendes Medium in Fragen der 99er-Schar aus der Taufe gehoben.

Nachdem dem Initiator und Gründer, Thomas M. Fiedler, von TI-Österreich kooperative Unterstützung in Form von Informationen und Literatur zugesichert war, fand sich innerhalb kürzester Zeit ein sehr starker Mitarbeiterkreis. Das Redaktionsteam versucht nicht nur Grundlagen, Tips und Tricks für die Arbeit am TI99 zu bieten, sondern vor allem komplexes Spezialwissen in klarer, verständlicher Form und in aufbauender Weise zu vermitteln.

So finden sich in den Journalen ein BASIC-Aufbaukurs (Dipl.-Ing. Tatzl), eine Assembler-Serie mit ausführlichen Programmbeispielen (Dipl.-Ing. Dr. Peschetz), Berichte über Hardware-Organisation und Selbstbaupläne (Mag. Karl Hagenbuchner), grafisch veranschaulicht sowie Modulbesprechungen, Kleinanzeigen, Grafikmöglichkeiten, Spielprogramm-Listings, Kriterien und viele Tips und Hinweise.

Tips und Tricks

Weiters besteht die Möglichkeit, eigene Software zum Verkauf (Kleinanzeigen) oder Tausch anzubieten (bzw. Programme und Kniffe zum Veröffentlichen zur Verfügung zu stellen) sowie aktiv durch Artikel, Besprechungen, Erfahrungsberichte und Kriterien am Inhalt mitzuarbeiten.

Clubmitglieder (das sind alles Abonnenten – es gibt keine Clubgebühr!) erhalten außerdem eine Mitgliedskarte (mit der sie bei einigen Firmen Einkaufsrabatte erhalten), Clubmitteilungsblätter und Softwarelisten. Außerdem besteht für jedes Clubmitglied die Möglichkeit, für Problemlösungen die Mithilfe der TI99-Redaktion in Anspruch zu nehmen.

Kontakte zu den amerikanischen und den Schweizer Clubs bestehen. Es wird an der Schaffung eines Clubtreffpunktes in Wien gearbeitet.

Auch nach dem Aussteigen von Texas Instruments aus dem Home-Computer-Geschäft wird das TI99-Journal die TI99-Anwender in unveränderter Weise weiter betreuen. So sind zum Beispiel seit Dezember 1983 ein BASIC- und Assembler-Spezial-Journal (Programmsammlungen) zu erhalten.

Interessenten (Mitarbeit, Erfahrungsaustausch, Abo. usw.) schreiben an:

-99 JOURNAL
Thomas M. Fiedler
1060 Wien
Mollardgasse 32/3
Telefon (0222) 5654042
(10–12 und 19–22 Uhr)

> Liebe Clubfans, diese Seite ist für Sie reserviert. Hier stehen Nachrichten, Termine und Porträts aus der Clubszene. Falls Sie etwas dazu beitragen wollen, senden sie uns Ihre Neuigkeiten zu.
> Ihre HC-Redaktion

CLUB ECKE

Scan: S. Höltgen


```
450  NI$ = NE$
460  PRINT " ": PRINT "   "; NU%(X + X1);" ";NE$;
470  IF A$ = CHR$ (13) THEN 82Ø
480  GOSUB 8Ø: IF A$ = CHR$ (13) THEN 82Ø
90   IF A$ = " " THEN GOSUB 8Ø: IF A$ "O" OR A$ =
     "G" OR A $ = "T" THEN 53Ø
500  IF A$ = " " THEN 49Ø
510  IF A$ = ":" THEN FX = Ø
520  GOTO 47Ø
530  A2$ = A$: IF A$ = "O" THEN GOSUB 8Ø: IF A$ =
     "N" THEN GOSUB 8Ø: IF A$ = " " THEN FX = 1:
     GOTO 47Ø
540  IF A2$ = "T" THEN GOSUB 8Ø: IF A$ = "H" THEN
     GOSUB 8Ø:IF A$ = "E" THEN GOSUB 8Ø: IF A$ =
     "N" THEN GOSUB 8Ø: IF A$ = " " THEN 6ØØ
550  A3$ = "": IF A2$ = "G" THEN GOSUB 8Ø: IF A$ =
     "O" THEN GOSUB 8Ø: IF A$ = "T" OR A$ = "S"
     THEN A3$ = A$: GOTO 57Ø
560  GOTO 47Ø
570  A4$ = "": GOSUB 8Ø:A4$ = A$: IF (A$ = "O"
     AND A3$ = "T") OR (A$ = "U" AND A3$ = "S")
     THEN GOSUB 8Ø: IF A3$ = "T" AND A$ = " " THEN
     6ØØ
580  IF A3$ = "S" AND A4$ = "U" AND A$ = "B" THEN
     GOSUB 8Ø: IF A$ = " " THEN 6ØØ
590  GOTO 47Ø
600  REM ROUT ZUR ZAHLENERMITTLUNG
610  N$ = "":FLAG = Ø
620  GET A$: IF A$ = "Ø" OR ( VAL (A$) > Ø AND A$
     <> "E") THEN FL = 1:N$ = N$ + A$: GOTO 62Ø
630  IF FL AND (A$ = " " OR A$ = "," OR A$ = CHR$
     (13) OR A$ = ":") THEN 66Ø
640  IF NOT FL THEN NE$ = NE$ + A$: GOTO 49Ø
650  NE$ = NE$ + N$ + A$: GOTO 47Ø
660  REM ZAHL GEFUNDEN
670  I2 = Ø: FOR I = Ø TO XM
680  IF VAL (N$) = NU%(I) THEN 76Ø
690  NEXT I
700  REM KEINE REFERENZ GEFUNDEN !!
710  PRINT " ACHTUNG! DIE ZEILE ";N$;" GIBT ES
     NICHT !!"
720  NI(X8,Ø) = NU%(X + X1):NI(X8,1) = VAL
     (NI$):NI(X8,2) = VAL (N$):X8 = X8 + 1
730  IF A$ = CHR$ (13) THEN FX = Ø
740  NE$ = NE$ + N$ + A$: IF FX THEN 6ØØ
750  GOTO 47Ø
760  REM ZEILENNUMMER GEFUNDEN
770  IF (I < XA) THEN 74Ø
780  IF A$ = CHR$ (13) THEN FX = Ø
790  I2 = ZN + (I - XA) * ZS
800  N$ = STR$ (I2): IF N$ = "Ø" THEN N$ = ""
810  GOTO 74Ø
820  B$(X1) = NE$
830  NEXT X1
840  PRINT D$
850  PRINT D$;"WRITE";FJ$
860  FOR I1 = Ø TO XN: PRINT B$(I1);:B$(I1) = "":
     NEXT I1
870  PRINT D$: PRINT D$
880  I1 = FRE (Ø)
890  IF S = 5 THEN 98Ø
900  NEXT X
910  GOTO 98Ø
920  REM ENDE
930  S = PEEK (222)
940  IF S = 5 THEN PRINT D$: PRINT D$;"CLOSE";FI$
950  IF S = 5 THEN 84Ø
960  PRINT " ": PRINT " *FATAL ERROR*ALL VALUES
     LOST !!"
970  PRINT " EXITING DUE TO ERROR NR ";S
980  PRINT : PRINT CHR$ (4);"CLOSE"
990  IF NI(Ø,Ø) = Ø THEN 1Ø6Ø
1000 PRINT "FOLGENDE REFERENZEN WURDEN NICHT":
     PRINT "GEFUNDEN !"
1010 PRINT "ZEILENNUMMER:"
1020 PRINT "ALT"; SPC( 6); "NEU"; SPCC 11);
     "REFERENZ"
1030 FOR X = Ø TO X8 - 1
1040 PRINT NT(X,Ø); SPC( 6);NI(X,1); SPC( 13);NI(X,2)
1050 NEXT X
1060 PRINT " *READY*"
1070 END
```

Poc-Man

Ein naher Verwandter von Pac-Man, der auf einem TI-99/4A von Geistern gejagt wird

Das Spiel läuft in drei Durchgängen ab, wobei der Bildschirm jedesmal die Farbe wechselt. Nach dem dritten Durchgang werden die Punktzahl und der Highscore angezeigt. Der Highscore und der Name des besten Spielers können dann auf Kassette gespeichert werden. Zum Lesen dieser Angaben von Kassette muß dem Programm die Zeile 100 GOSUB 3900 hinzugefügt werden.

Soll die Steuerung des Poc-Man statt mit Joystick über die Tastatur erfolgen, so sind folgende Änderungen nötig:

```
1510 CALL KEY (0,K,S)
1530 IF S=0 THEN 1510
1540 IF K=101 THEN 1660
1550 IF K=120 THEN 1720
1560 IF K=100 THEN 1780
1570 IF K=115 THEN 1840
```

Die Tasten E, S, D und X (Alpha-Lock-Taste nicht gedrückt) geben dann die Richtung an.

Eine Einführung zu Beginn des Programmablaufs erklärt das Spiel noch einmal in allen Einzelheiten. *Klaus Brackmann*

Scan: S. Höltgen


```
110 CALL CLEAR
120 CALL SCREEN(14)
130 PRINT "       ###########"
140 PRINT "       #         #"
150 PRINT "       # poc man #"
160 PRINT "       #         #"
170 PRINT "       ###########"
::::::::
180 FOR P=1 TO 1000
190 NEXT P
200 CALL CLEAR
210 CALL SCREEN(4)
220 PRINT "kennst du die spie
lregeln <j><n>?"
230 CALL KEY(0,K,S)
240 IF S=0 THEN 230
250 IF K=106 THEN 550
260 PRINT "s p i e l r e g e
l n"
270 PRINT "auf dem spielfeld
erscheinen: "
280 PRINT " 100 mal futter(bl
au)"
290 PRINT " 4 mal superfutter
(weiss)"
300 PRINT " 1 mal krone(schat
z)"
310 PRINT " 4 gespenster"
320 PRINT " 1 poc man"
330 PRINT "du musst mit hilfe
 des joy  stick versuchen,so
viele    futter wie mo
eglich  zu     fressen."
340 PRINT
350 PRINT "versuche auch den
schatz zu rauben.aber lass di
ch nicht erwischen."
360 PRINT
370 PRINT "weiter mit <a>"
380 CALL KEY(0,K,S)
390 IF S=0 THEN 380
400 IF K=97 THEN 420
410 GOTO 370
420 CALL CLEAR
430 PRINT
440 PRINT "wenn du alle punkt
e gefres  sen hast,kannst du
durch denausgang unten
 weglaufen."
450 PRINT "du kannst natuerli
ch auch   schon frueher gehen
."
460 PRINT "den schatz kannst
du aber   erst holen,wenn ein
 gespenstnachgesehen h
at,ob er noch  da ist.";
470 PRINT "er vergisst die tu
er hinter sich zu schliessen.
"
480 PRINT "derenk dran:jeder
schritt   kostet kraft."
490 PRINT "alles verstanden?
<j><n>"
500 CALL KEY(0,K,S)
510 IF S=0 THEN 500
520 IF K=106 THEN 550
530 IF K=110 THEN 200
540 GOTO 490
550 BONUS=0
560 GES=0
570 POC=3
580 K=0
590 FR=0
600 CALL CLEAR
610 H4=28
620 H3=22
630 H2=17
640 VTAB=2
650 HTAB=3
660 H1=8
670 G4=20
680 G3=16
690 G2=12
700 G1=6
710 REM  ZEICHENCODES
720 CALL CLEAR
730 GOSUB 3810
740 CALL SCREEN(SC)
750 CALL CHAR(159,"FFFFFFFFFF
FFFFFF")
760 CALL CHAR(157,"FFFFFFFFFF
FFFFFF")
770 CALL CHAR(151,"FFFFFFFFFF
FFFFFF")
780 CALL CHAR(143,"3C367E7EF0
F03C3C")
790 CALL CHAR(142,"3C7EDBFFFF
FFFF55")
800 CALL CHAR(141,"00003C3C3C
3C0000")
810 CALL CHAR(135,"00003C3C3C
3C0000")
820 CALL CHAR(64,"0099999999F
FFF00")
830 CALL COLOR(16,2,2)
840 CALL COLOR(15,13,1)
850 CALL COLOR(14,16,1)
860 CALL COLOR(13,5,1)
870 CALL COLOR(5,11,1)
880 REM  SPIELFELDAUFBAU
890 CALL HCHAR(1,1,159,32)
900 CALL HCHAR(23,1,159,14)
910 CALL HCHAR(23,17,159,16)
920 CALL HCHAR(24,1,151,32)
930 CALL VCHAR(2,1,159,21)
940 CALL VCHAR(2,32,159,21)
950 CALL HCHAR(10,2,159,5)
960 CALL HCHAR(16,2,159,5)
970 CALL HCHAR(4,2,159,5)
980 CALL HCHAR(5,14,159,7)
990 CALL HCHAR(9,14,159,5)
1000 CALL HCHAR(13,14,157,5)
1010 CALL HCHAR(20,13,159,7)
1020 CALL HCHAR(4,27,159,5)
1030 CALL HCHAR(10,27,159,5)
1040 CALL HCHAR(16,27,159,5)
1050 CALL VCHAR(2,10,159,4)
1060 CALL VCHAR(11,9,159,4)
1070 CALL VCHAR(19,9,159,4)
1080 CALL VCHAR(10,14,157,3)
1090 CALL VCHAR(6,20,159,2)
1100 CALL VCHAR(10,18,157,3)
1110 CALL VCHAR(15,16,159,3)
1120 CALL VCHAR(2,23,159,4)
1130 CALL VCHAR(12,24,159,4)
1140 CALL VCHAR(19,23,159,4)
1150 CALL HCHAR(24,5,143,(POC
-1))
1160 GOSUB 1280
1170 REM  KRAFTFUTTER SETZEN
1180 RANDOMIZE
1190 FOR I=1 TO 4
1200 A=INT((21-3+1)*RND)+3
1210 B=INT((30-3+1)*RND)+3
1220 CALL GCHAR(A,B,X)
1230 IF X=159 THEN 1200
1240 CALL HCHAR(A,B,141)
1250 CALL SOUND(-100,500,0)
1260 NEXT I
1270 GOTO 1400
1280 REM  FUTTER SETZEN
1290 FOR I=1 TO 100
1300 RANDOMIZE
1310 C=INT((21-3+1)*RND)+3
1320 D=INT((30-3+1)*RND)+3
1330 CALL GCHAR(C,D,X)
1340 IF X=159 THEN 1300
1350 IF X=157 THEN 1300
1360 CALL HCHAR(C,D,135)
1370 CALL SOUND(-10,1000,0)
1380 NEXT I
1390 RETURN
1400 REM   POC MAN SETZEN
1410 CALL HCHAR(11,16,64)
1420 CALL SOUND(-100,110,0)
1430 E=3
1440 F=8
1450 CALL SOUND(500,330,0)
1460 CALL HCHAR(E,F,32)
1470 CALL HCHAR(E,F,143)
1480 GOTO 1490
1490 REM   BEWEGEN POC-MAN
1500 BEW=0
1510 REM
1520 IF BEW=5 THEN 1900
1530 CALL JOYST(1,X,Y)
1540 IF Y=4 THEN 1660
1550 IF Y=-4 THEN 1720
1560 IF X=4 THEN 1780
1570 IF X=-4 THEN 1840
1580 GOTO 1510
1590 REM   BEW POC MAN
1600 CALL GCHAR(E,F,X)
1610 IF X<>32 THEN 2770
1620 CALL HCHAR(E,F,143)
1630 CALL SOUND(-100,-4,0)
1640 BEW=BEW+1
1650 RETURN
1660 REM  NACH OBEN
1670 K=K-1
1680 CALL HCHAR(E,F,32)
1690 E=E-1
1700 GOSUB 1590
1710 GOTO 1510
1720 REM  NACH UNTEN
1730 K=K-1
1740 CALL HCHAR(E,F,32)
1750 E=E+1
1760 GOSUB 1590
1770 GOTO 1510
1780 REM  NACH RECHTS
1790 CALL HCHAR(E,F,32)
1800 K=K-1
1810 F=F+1
1820 GOSUB 1590
1830 GOTO 1510
1840 REM  NACH LINKS
1850 K=K-1
1860 CALL HCHAR(E,F,32)
1870 F=F-1
1880 GOSUB 1590
1890 GOTO 1510
1900 REM  GESPENSTER SETZEN
1910 CALL HCHAR(G1,H1,32)
1920 GOSUB 2710
1930 H1=H
1940 G1=G
1950 CALL GCHAR(G1,H1,X)
1960 IF X=159 THEN 1920
1970 CALL GCHAR(G1,H1,X)
1980 IF X=143 THEN 2320
1990 CALL HCHAR(G1,H1,142)
2000 CALL HCHAR(G2,H2,32)
2010 GOSUB 2710
2020 G2=G
2030 H2=H
2040 CALL GCHAR(G2,H2,X)
2050 IF X=159 THEN 2010
2060 CALL GCHAR(G2,H2,X)
2070 IF X=143 THEN 2410
2080 CALL HCHAR(G2,H2,142)
2090 CALL HCHAR(G3,H3,32)
2100 GOSUB 2710
2110 G3=G
2120 H3=H
2130 CALL GCHAR(G3,H3,X)
2140 IF X=159 THEN 2100
2150 CALL GCHAR(G3,H3,X)
2160 IF X=143 THEN 2510
2170 CALL HCHAR(G3,H3,142)
2180 GOSUB 2710
2190 G4=G
2200 H4=H
2210 IF POC>1 THEN 2250
2220 CALL SCREEN(4)
2230 FOR DEL=1 TO 50
2240 NEXT DEL
2250 CALL SCREEN(SC)
2260 CALL GCHAR(G4,H4,X)
```

Scan: S. Höltgen


```
2270 IF X=159 THEN 2180
2280 CALL GCHAR(G4,H4,X)
2290 IF X=143 THEN 2610
2300 CALL HCHAR(G4,H4,142)
2310 GOTO 1510
2320 FOR I=1 TO 11
2330 CALL SOUND(-100,1500,0)
2340 CALL HCHAR(G1,G2,142)
2350 FOR PA=1 TO 50
2360 NEXT PA
2370 CALL HCHAR(G1,G2,143)
2380 NEXT I
2390 FOR PA=1 TO 50
2400 NEXT PA
2410 FOR I=1 TO 11
2420 CALL SOUND(-100,1500,0)
2430 CALL HCHAR(G2,H2,142)
2440 FOR PA=1 TO 100
2450 NEXT PA
2460 CALL HCHAR(G2,H2,143)
2470 FOR PA=1 TO 50
2480 NEXT PA
2490 NEXT I
2500 GOTO 2930
2510 FOR I=1 TO 11
2520 CALL SOUND(-100,1500,0)
2530 CALL HCHAR(G3,H3,142)
2540 FOR PA=1 TO 50
2550 NEXT PA
2560 CALL HCHAR(G3,H3,143)
2570 FOR PA=1 TO 50
2580 NEXT PA
2590 NEXT I
2600 GOTO 2930
2610 FOR I=1 TO 11
2620 CALL SOUND(-100,1500,0)
2630 CALL HCHAR(G4,H4,142)
2640 FOR PA=1 TO 50
2650 NEXT PA
2660 CALL HCHAR(G4,H4,143)
2670 FOR PA=1 TO 50
2680 NEXT PA
2690 NEXT I
2700 GOTO 2930
2710 REM  BEWEGEN GESP
2720 RANDOMIZE
2730 G=INT((21-3+1)*RND)+3
2740 H=INT((30-3+1)*RND)+3
2750 BEW=0
2760 RETURN
2770 REM  AUFLAUF-ERKENNUNG
2780 IF X=159 THEN 2860
2790 IF X=157 THEN 2860
2800 IF X=151 THEN 3170
2810 IF X=142 THEN 2930
2820 IF X=141 THEN 3110
2830 IF X=135 THEN 3140
2840 IF X=64 THEN 3040
2850 END
2860 REM  AUFLAUF RAND
2870 CALL SCREEN(4)
2880 DRUCK$="du kannst nicht
mit dem kopf  durch die wand"
2890 GOSUB 3620
2900 FOR PA=1 TO 1000
2910 NEXT PA
2920 GOTO 3280
2930 REM  auflauf gespenst
2940 CALL SCREEN(4)
2950 DRUCK$="friede deiner as
che!"
2960 GOSUB 3620
2970 FOR NN=1 TO 500
2980 NEXT NN
2990 DRUCK$="du bist gefresse
n!"
3000 GOSUB 3620
3010 FOR PA=1 TO 500
3020 NEXT PA
3030 GOTO 3320
3040 REM  SCHATZ GEFUNDEN
3050 FOR I=1 TO 8
3060 TONE=TONE+110
3070 CALL SOUND(50,TONE,0)
3080 NEXT I
3090 K=K+100
3100 GOTO 1620
3110 REM  kraftfutter
3120 K=K+50
3130 GOTO 1620
3140 REM  futter fassen
3150 K=K+5
3160 GOTO 1620
3170 REM  fluchtweg
3180 CALL SCREEN(4)
3190 DRUCK$="du hast den ausg
ang benutzt"
3200 GOSUB 3620
3210 FOR PA=1 TO 500
3220 NEXT PA
3230 BONUS=BONUS+50
3240 GOTO 3290
3250 DRUCK$="f e i g l i n g"
3260 GOSUB 3620
3270 CALL SCREEN(4)
3280 DRUCK$="leider kein bonu
s"
3290 REM  endroutine
3300 POC=POC-1
3310 IF POC>0 THEN 710
3320 FOR PA=1 TO 1000
3330 NEXT PA
3340 DRUCK$="punktzahl: "&STR
$(K)
3350 GOSUB 3620
3360 DRUCK$="bonus     : "&STR
$(BONUS)
3370 GOSUB 3620
3380 GES=K+BONUS
3390 DRUCK$="gesamt    : "&STR
$(GES)
3400 GOSUB 3620
3410 IF GES<HS THEN 3460
3420 DRUCK$="gib deinen namen
 ein "
3430 GOSUB 3620
3440 INPUT NAME$
3450 HS=GES
3460 DRUCK$="higthsc.  : "&STR
$(HS)
3470 GOSUB 3620
3480 DRUCK$="spieler   : "&NA
ME$
3490 GOSUB 3620
3500 DRUCK$="noch einmal?"
3510 GOSUB 3620
3520 DRUCK$="ja,roten knopf d
ruecken"
3530 GOSUB 3620
3540 CALL KEY(1,K,S)
3550 IF S=0 THEN 3570
3560 IF K=18 THEN 550
3570 CALL JOYST(1,X,Y)
3580 IF Y=4 THEN 3600
3590 GOTO 3540
3600 GOTO 3940
3610 BONUS=BONUS+100
3620 REM  printroutine
3630 FOR DRI=1 TO LEN(DRUCK$)
3640 CODE=ASC(SEG$(DRUCK$,DRI
,1))
3650 GOSUB 3690
3660 NEXT DRI
3670 GOSUB 3740
3680 RETURN
3690 REM
3700 CALL HCHAR(VTAB,HTAB,COD
E)
3710 HTAB=HTAB+1
3720 IF HTAB<32 THEN 3790
3730 REM
3740 VTAB=VTAB+1
3750 HTAB=2
3760 IF VTAB<25 THEN 3790
3770 VTAB=24
3780 PRINT
3790 RETURN
3800 END
3810 IF POC=3 THEN 3840
3820 IF POC=2 THEN 3860
3830 IF POC=1 THEN 3880
3840 SC=14
3850 RETURN
3860 SC=15
3870 RETURN
3880 SC=2
3890 RETURN
3900 OPEN #1:"CS1",FIXED 192,
INPUT ,INTERNAL
3910 INPUT #1:NAME$,HS
3920 CLOSE #1
3930 RETURN
3940 OPEN #1:"CS1",SEQUENTIAL
,FIXED 192,OUTPUT,INTERNAL
3950 PRINT #1:NAME$,HS
3960 CLOSE #1
3970 END
```

Scan: S. Höltgen

Die zehn einfachsten Programme

„In der Kürze liegt die Würze." Auch kleine Programme können leistungsfähig sein. Dem blutigen Laien vermitteln einfache Programme oft ein erstes Verständnis dafür, wie Computer in der Sprache BASIC angesprochen werden wollen

Rechenlehrer

```
10 PRINT "MALNEHMEN
   PAUKEN"
20 LET Z1=INT(RND
   *100)
30 LET Z2=INT (RND
   *100)
40 PRINT "WAS IST ";
   Z1;"*";Z2;"?"
50 INPUT A
60 IF A=Z1*Z2 THEN
   PRINT "RICHTIG"
70 IF A<>Z1*Z2 THEN
   PRINT "FALSCH"
80 GOTO 20
```

Zwei Zahlen zwischen 1 und 100 sollen multipliziert werden. Das Programm stellt die Aufgaben und sagt, ob man richtig oder falsch gerechnet hat.

Die Zeilen 20 und 30 wählen die beiden Zahlen aus; die Zeile 40 stellt die Frage. Nachdem in Zeile 50 die Eingabe entgegengenommen wurde, wird das Produkt mit der Eingabe verglichen und "RICHTIG" oder "FALSCH" ausgedruckt. In Zeile 80 erfolgt ein Rücksprung zur nächsten Aufgabenstellung.

Summe der ersten N natürlichen Zahlen

```
10 PRINT "SUMME DER
   ERSTEN N
   NATUERLICHEN
   ZAHLEN"
20 PRINT "BIS ZU
   WELCHER ZAHL?"
30 INPUT N
40 LET S=0
50 FOR I=1 TO N
60 LET S=S+I
70 NEXT I
80 PRINT S
```

In der Zeile 40 wird die Summe zunächst auf Null gesetzt. Sie vergrößert sich in den Zeilen 50 bis 70, indem jeweils die nächste Zahl addiert wird. Die Zeile 80 druckt die Summe.

Code knacken

```
10 PRINT "ZAHLEN
   RATEN"
20 INPUT A
30 PRINT "WAS TIPPST
   DU?"
40 INPUT B
50 IF A=B THEN GOTO
   100
60 IF A>B THEN PRINT
   "ZU KLEIN"
70 IF A<B THEN PRINT
   "ZU GROSS"
80 PRINT "VERSUCHE
   ES NOCH EINMAL"
90 GOTO 30
100 PRINT "RICHTIG"
110 GOTO 10
```

Der Home-Computer teilt mit, wie man mit seinem Tip liegt. Die Geheimzahl wird in Zeile 20 von einer zweiten Person eingegeben. Zeile 40 erwartet einen Tip, und die Zeilen 50 bis 70 stellen fest, ob man richtig, drüber oder drunter liegt.

Scan: S. Höltgen

Lottozahlen

```
10 PRINT "LOTTO-
   ZAHLEN"
20 DIM L(7)
30 FOR I=2 TO 7
40 LET L(I)=INT(RND
   *49+1)
50 FOR J=1 TO I-1
60 IF L(I)=L(J) THEN
   GOTO 40
70 NEXT J
80 PRINT L(I)
90 NEXT I
```

Vielleicht verhilft Ihnen Ihr Home-Computer zum großen Glück: Dieses Programm schlägt Ihnen Lotto-Tips vor. In Zeile 40 wird eine Zahl zwischen 1 und 49 zufällig bestimmt. Durch die Zeilen 50 bis 70 wird vermieden, daß eine Zahl zweimal gezogen wird.

Begrüßung

```
10 INPUT N$
20 PRINT "GUTEN TAG
   HERR ";N$
30 GOTO 10
```

Dieses Programm zeigt, wie freundlich Ihr Home-Computer sein kann. Jedem, der seinen Namen eingibt, wünscht er einen guten Tag.

Zinsen

```
10 PRINT "EINGABE
   KAPITAL"
20 INPUT K
30 PRINT "DAUER
   ZINSSATZ ZINSEN"
40 FOR I=1 TO 10
50 FOR P=1 TO 10
60 LET Z=K*(1+P/
   100)↑I
70 PRINT I;TAB 7;P;TAB
   16;Z
80 NEXT P
90 NEXT I
```

Welchen Einfluß haben Laufzeit und Zinsrate auf das Anwachsen eines Kapitals? Das Programm druckt eine Tabelle aus, in der die Zinsrate zwischen 1 % und 10 % und die Laufzeit zwischen einem und zehn Jahren variieren.

Pythagoras

```
10 PRINT "SATZ DES
   PYTHAGORAS"
20 PRINT "EINGABE DER
   KATHETENLAENGEN"
30 INPUT A
40 INPUT B
50 LET C=SQR(A*A+
   B*B)
60 PRINT "LAENGE DER
   HYPOTHENUSE= ";C
70 GOTO 20
```

Nach der Eingabe der beiden Kathetenlängen eines rechtwinkligen Dreiecks in den Zeilen 30 und 40 wird in der Zeile 50 die Länge der Hypotenuse berechnet und in Zeile 60 ausgegeben.

Winkel-Umrechnung vom Gradmaß ins Bogenmaß

```
10 PRINT
   "UMRECHNUNG
   GRADMASS IN
   BOGENMASS"
20 PRINT "EINGABE
   EINES WERTES IM
   GRADMASS"
30 INPUT G
40 LET B=3.14159*G/180
50 PRINT
   "GRADMASS=";G,
   "BOGENMASS=";B
60 GOTO 20
```

Bei den meisten Home-Computern erwarten die trigonometrischen Funktionen (sin, cos, tan, cot usw.), daß ein Winkel im Bogenmaß angegeben wird. Dieses Programm übernimmt die Umrechnung einer eingegebenen Zahl vom gewohnten Gradmaß ins Bogenmaß. In Zeile 40 wird die eingegebene Zahl mit der Zahl Pi multipliziert und das Ergebnis durch 180 geteilt (stattdessen könnte man auch mit dem Doppelten von Pi multiplizieren und dann durch 360 teilen, was natürlich zum gleichen Ergebnis führen würde).

Größter gemeinsamer Teiler

```
  5 PRINT "GGT VON
    DREI ZAHLEN"
 10 PRINT "EINGABE
    VON DREI ZAHLEN"
 20 INPUT A
 22 INPUT B
 24 INPUT C
 30 LET X=A
 40 LET Y=B
 50 GOSUB 200
 60 LET X=G
 70 LET Y=C
 80 GOSUB 200
 90 PRINT TAB 13;"GGT"
 95 PRINT A;TAB
    5;B;TAB 9;C;TAB
    13;G
100 STOP
200 LET Q=INT(X/Y)
210 LET R=X-Q* Y
220 IF R=0 THEN GOTO
    300
230 LET X=Y
240 LET Y=R
250 GOTO 200
300 LET G=Y
310 RETURN
```

Das Programm ruft zweimal ein Unterprogramm auf. Dieses Unterprogramm steht in den Zeilen 200 bis 310 und kann den größten gemeinsamen Teiler (ggT) von zwei Zahlen berechnen. Zur Bildung des ggT der ersten beiden Zahlen wird es in der Zeile 50 zum ersten Mal aufgerufen. Danach soll der ggT des Ergebnisses und der dritten Zahl berechnet werden; dazu wird das Interprogramm in der Zeile 80 erneut aufgerufen. Schließlich werden alle drei Zahlen und der ggT in der Zeile 95 ausgedruckt.

Würfel

```
 10 PRINT "STATISTIK
    DER SUMME VON
    2 WUERFELN"
 20 DIM H(12)
 30 FOR I=1 TO 100
 40 LET W1=INT(RND
    *6+1)
 50 LET W2=INT(RND
    *6+1)
 60 LET A=W1+W2
 70 LET H(A)=H(A)+1
 80 NEXT I
100 PRINT "AUGEN
    ANZAHL"
110 FOR I=2 TO 12
120 PRINT I;TAB 7;H(I)
130 NEXT I
```

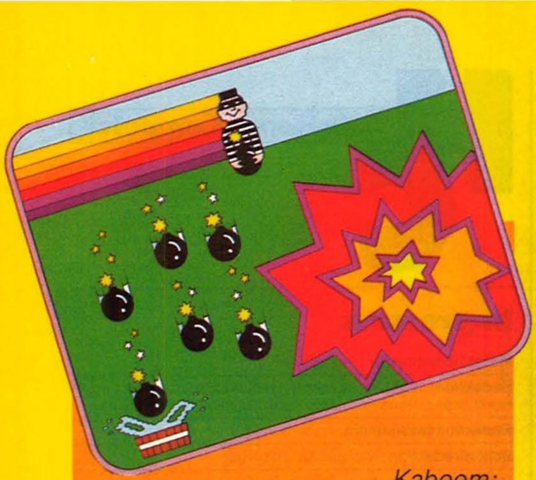
Als Augensumme von zwei Würfeln tritt ja zum Beispiel die Zahl 12 häufiger auf als die Zahl 3. Dieses Programm simuliert 100 Würfe und gibt danach eine Statistik über die Häufigkeit der einzelnen Augensummen aus.
In den Zeilen 40 und 50 werden ganze Zahlen zwischen 1 und 6 für die beiden Würfel zufällig ausgewählt. Die Summe wird in Zeile 60 errechnet. Die Häufigkeit für die erreichte Augensumme wird in Zeile 70 um eins erhöht. Die Statistik wird schließlich ab Zeile 100 ausgegeben.

Hinweis

Einige Home-Computer verlangen kleine Änderungen in den Programmen bei den IF-Anweisungen, R*N*D, TAB oder der Stringvariablen *N*$ im Programm „Begrüßung“.

br

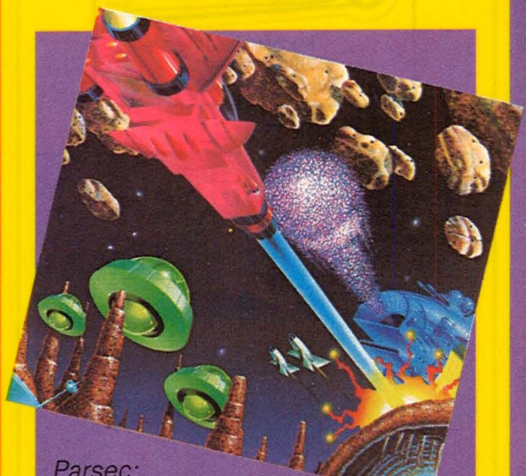
Scan: S. Höltgen



Kaboom:

Der Bombenwerfer ist los

Die Gegner von Kriegsspielen werden schon wieder den Zeigefinger heben, wenn von Bomben die Rede ist. Doch bei „Kaboom" führt sie das Foto auf der Verpackung ganz schön an der Nase herum. Der „verrückte Bombenwerfer" entpuppt sich auf dem Bildschirm als eine sehr witzige Figur, die dem Spieler das Reagieren ganz schön schwer macht. Und hier sind wir bereits beim Zweck dieses Spiels: Das Reaktionsvermögen wird auf eine harte Probe gestellt. Jedenfalls herrscht im wahrsten Sinne des Wortes „Bombenstimmung" auf dem Bildschirm. Also: Da tritt der Hauptdarsteller, der verrückte Bombenwerfer, auf. Er schleudert seine hochexplosiven Bomben vom Dach eines Hauses nach unten. Der Spieler bekommt auf dem Bildschirm drei Wasserwerfer, die er so geschickt bewegen muß, daß die Bomben damit aufgefangen werden können. Wehe, das gelingt nicht. Jedesmal, wenn nämlich eine Bombe danebenfällt, explodieren die auf dem Bildschirm verbliebenen Bomben, und der Spieler verliert seinen unteren Wassereimer. Der „verrückte Bombenwerfer" kommentiert das Mißgeschick mit breitem Grinsen, um damit seiner Freude Ausdruck zu verleihen. Doch auch wenn sich der Spieler geschickt verhält und die Bomben auffängt, muß er sich weiter mit dem Verrückten herumschlagen. Je mehr Bomben aufgefangen werden, um so schneller wirft der Verrückte weitere hinterher.
Schwierigkeitsgrad: Mit der Option-Taste können drei verschiedene Schwierigkeitsgrade gewählt werden.
Spieler: 1 oder 2
Unser Testurteil: Ein Spiel, das trotz der „Bombe" auch für den Familienkreis geeignet ist. Wir meinen, den Programmierern ist mit Kaboom ein sehr spannendes Spiel gelungen, das auch auf Dauer nicht langweilig wird. Wer viele Computerspiele kennt, der wird in Kaboom ein altes Spielmuster erkennen, das ähnlich auch bei Weltraumspielen zu beobachten ist – der Angriff von oben nach unten durch herabfallende Gegenstände oder Waffen. Nach unserem Test möchten wir Kaboom auch als Partyspaß empfehlen. Da zwei Spieler gleichzeitig die Joysticks beziehungsweise Drehregler bedienen können, ergibt sich jeweils ein spannender Zweikampf, der nach dem K.-o.-System zum spannenden Ausscheidungskampf umfunktioniert werden kann. Fazit: Ein Spiel, das die Reaktionsfähigkeit fördert.
Hersteller: Activision
Geeignet für: Atari 400, 600 XL und 800
Preis: 139 Mark



Parsec:

Raumschiff in Gefahr

Da fliegt ein Raumschiff friedlich über den Planeten. Die Besatzung plaudert über das letzte Wochenende und entspannt sich bei einer Tasse Kaffee. Da tauchen plötzlich fremde Luftschiffe auf. Jetzt kann nur noch die Laserkanone an Bord helfen. Sie kann durch gezielten Einsatz die Angreifer zum Absturz bringen. Vor lauter Angst läßt der Mann an der Laserkanone den Finger ständig auf dem Auslöseknopf für die Laserkanone. Da erscheint die Anzeige „Überhitzt", und aus ist es mit der Verteidigung durch Laser. Das Raumschiff hat alle Raffinessen, die die moderne Elektronik heute bietet, an Bord. Zum Beispiel warnt eine synthetische Stimme vor fremden Raumschiffen, die sich nähern, und macht auch auf die Situation beim Treibstoffverbrauch aufmerksam. Geht der Treibstoff zu Ende, muß ein schwieriges Flugmanöver durch den Betankungstunnel gemeistert werden. Gefahr droht auch durch einen Astroidengürtel, durch den das Raumschiff seine Flugbahn freischießen muß. Besonders kritisch sind die Angriffswellen von sieben verschiedenen fremden Raumschiffen. Achtung auf die Flughöhe. Wir sind bei unseren Tests immer wieder abgeschossen worden, weil wir nicht bedacht haben, daß die fremden Flugzeuge tückischerweise unter dem Rumpf ihre Bordkanonen abfeuern. Ein Spiele-Freak erwartet den gegnerischen Schuß aus Erfahrung in Höhe der Tragflächen.
Schwierigkeitsgrad: drei Spielstufen
Spieler: 1
Unser Testurteil: Nach unserer Meinung eines der raffiniertesten Spiele aus dem Programm von Texas Instruments. Es wäre schon der Anlaß, sich einen TI 99/4A zu kaufen, um das Programmodul abspielen zu können. Da kommt man so richtig in das Schwitzen. Und höllisch aufpassen muß man dazu noch. Geschicklichkeit und strategische Fähigkeiten sind gefragt. Die ausführliche Anleitung erklärt die verschiedenen Spielstufen. Gefallen hat uns auch der Anzeigemodus. Ob der Treibstoff zu Ende geht, die Laserkanone überhitzt ist oder das Raumschiff von feindlichen Flugzeugen getroffen wurde, alles wird angezeigt – mit Worten in englischer Sprache. Der Antrieb des Raumschiffes ist auf drei Stufen umschaltbar, was wiederum das Spiel spannender gestalten läßt.
Hersteller: Texas Instruments
Geeignet für: TI 99/4A
Preis: 148 Mark

Fusion

Diese beiden Maschinenprogramme ermöglichen, zu einem ZX 81-BASIC-Programm weitere hinzuzuladen, ohne daß das vorherige jeweils gelöscht wird; sie wurden für die 16-KByte-Erweiterung geschrieben

Das Maschinenprogramm VER verschiebt ein BASIC-Programm A mit allen Variablen hinter RAMTOP; das zweite FUS koppelt dieses Programm A an ein im BASIC-Bereich liegendes Programm B an (wobei die Variablen von B gelöscht werden). Dabei ist natürlich darauf zu achten, daß die kleinste Zeilennummer von A größer ist als die größte von B. Das Fusionsprogramm läuft nur im FAST-Modus; versucht man, es im SLOW-Modus zu starten, erscheint die Fehlermeldung X/O.

Vor dem Starten des Programmes mit RUN müssen die beiden Kommandos
POKE 16389,95
NEW
eingegeben werden.

Helmut Linneweber

```
POKE 16389,95
NEW

  30 REM VERSCHIEBUNG UND FUSION
 VON BASIC PROGRAMMEN BEIM ZX 81
  40 REM  © HELMUT LINNEWEBER
  41 REM    RAUCHLOCHWEG 8
  42 REM    53 BONN 3
  43 REM
  45 REM VER
  46 LET A=PEEK 16388+256*PEEK 1
6389
  47 PRINT "VER IST DURCH RAND U
SR ";A;" AUFZURUFEN."
  50 LET M$="2A1440117D40ED52444
D217B407123702A1440110940ED52444
DEB110960EDB0C9"
  60 GOSUB 100
  65 REM FUS
  67 PRINT "FUS IST DURCH (FAST)
 RAND USR ";A;" AUFZURUFEN."
  70 LET M$="2100403620EB213B403
EFF9ECB7FCA2F00EB36FF217B40ED5B0
C40732372 2A7B60444D217D60ED5B7B4
0EDB02A7B40117D40ED52444D2A0C600
9110C40EB7323722A1060091110 40EB7
323722A14600911 1440EB7323722A1A6
009111A40EB7323722A1C6009111C40E
B7323722A0E6009110E40EB7323722A1
66009111640EB7323722A29600911294
0EB732372C9"
  80 GOSUB 100
  85 STOP
  90 SAVE "V█"
  91 RUN
 100 LET M$=M$+"*"
 105 LET L=16*(CODE M$(1)-28)+CO
DE M$(2)-28
 110 POKE A,L
 111 LET M$=M$(3 TO )
 113 LET A=A+1
 115 IF M$(1)="*" THEN RETURN
 116 GOTO 105
```

Pongo

Pongo soll Vasen einsammeln und klettert dazu über Leitern von Ebene zu Ebene. Verschiedenartige Wesen versuchen ihn daran zu hindern. Ein gelungenes Spiel für den TI-99/4A mit Extended-BASIC

Nach dem Programmstart mit RUN wird man gefragt, ob man eine Spielanleitung haben möchte oder nicht. Beantwortet man die Frage mit „J", so erhält man folgende Spielanleitung:

Ziel des Spieles ist es, Pongo auf eine der beiden oberen Leitern zu führen und dabei so viele Vasen wie möglich zu sammeln. Pongo kann eine der Leitern nur dann unbeschadet hinaufsteigen, wenn er mindestens sieben Vasen auf seinem Weg nach oben gesammelt hat. Betritt Pongo eine der zwei oberen Leitern, ohne mindestens sieben Vasen zu haben, so fällt er vom Gerüst. Pongo fällt ebenfalls vom Gerüst, wenn er auf einer Leiter zur Seite geht, über oder in die Nähe eines Loches gerät oder den Bildschirm an den Seiten verläßt. Er fällt ebenfalls hinunter, wenn er von einem Wesen auf dem Gerüst berührt wird.

Steuerung

Pongo bewegt sich mit
„," – nach links
„." – nach rechts
„A" – die Leiter hinauf
„Z" – die Leiter hinunter

Zu Beginn stehen dem Spieler jeweils insgesamt fünf Pongos zur Verfügung. Alle 5000 Punkte gibt es einen zusätzlichen Pongo als Bonus.

Punktanzeige

links – Rundenzahl
Mitte – Anzahl der gesammelten Vasen
rechts – Punkte

Punktwertung

Für jede gesammelte Vase erhält man 30+5*Rundenzahl Punkte. Für das Erreichen einer der beiden

Scan: S. Höltgen

oberen Leitern mit mindestens sieben Vasen gibt es 500+10*Rundenzahl Punkte. Wenn es Pongo gelingt, in einer Runde alle zehn Vasen zu sammeln, so erhält man nochmals 200+50* Rundenzahl Punkte.

Nach dem Spielende drückt man irgendeine Taste, um ein neues Spiel zu beginnen. *Volker Becker*

```
9 DATA 1,5,1,27,5,10,5,22,9,
3,9,29,13,13,13,19,17,2,17,2
3,21,7,21,27
10 CALL CLEAR :: CALL CHARSE
T :: CALL SCREEN(11)
11 GOTO 20 :: Z:T :: S :: RU
NDE :: PUNKTE :: PONGO :: BO
NUS :: VASE :: I :: A ::
 B :: P :: E1 :: E2 :: E3 ::
 J :: MOT :: ST :: R :: C ::
 X :: Y :: BP
12 CALL CHAR :: CALL MAGNIFY
 :: CALL SPRITE :: CALL KEY
:: CALL EINLEITUNG :: CA
LL COLOR :: CALL DELSPRITE :
: CALL HCHAR :: CALL GCHAR :
: CALL STAND
13 CALL SOUND :: CALL PONGOA
NZ :: CALL COINC :: CALL MOT
ION :: CALL POSITION ::
CALL PATTERN :: CALL VCHAR :
: CALL MELODY :: CALL LOCATE
 :: CALL MELODY2
14 CALL MELODY3 :: !@P-
20 CALL CHAR(128,"000103061F
1F0703113F3F130707070F10F0E0
F0F0F0E0C080E0E0F0FB3B3E
78")
30 CALL CHAR(64,"3C4299A1A19
9423C")
40 DISPLAY AT(1,10):"P O N G
 O" :: DISPLAY AT(2,10):"===
======" :: CALL MAGNIFY(
4):: CALL SPRITE(#1,128,14,2
4,106)
50 DISPLAY AT(24,1):"     @ 1
983 VOLKER BECKER    "
60 DISPLAY AT(12,1):"       W
OLLEN SIE EINE            S
PIELANLEITUNG ?
      (J / N)"
70 CALL KEY(5,T,S):: IF T=74
 THEN 80 ELSE IF T=78 THEN 9
0 ELSE 70
80 CALL EINLEITUNG
90 CALL CLEAR
95 RUNDE=0 :: PUNKTE=0 :: PO
NGO=5 :: BONUS=0
100 ON ERROR 690
110 CALL COLOR(2,8,2,3,10,2,
4,10,2)
120 CALL COLOR(5,10,2,6,10,2
,7,10,2,8,10,2)
130 CALL CHAR(100,"001B1F0F0
F0FC7F3373F1F1F1F0F060E00D8F
8F0F0F0E3CFECFCF8F8F8F06
070")
140 CALL CHAR(104,"000000000
000060F86F6FEFF1F9EFEF800000
0000000000303078FCF6FEFE7
E3B")
150 CALL CHAR(96,"000000183C
2EFE1F0100000000000100000000
000000000084C2F2E6CC98F0
E0")
160 CALL CHAR(36,"0000000000
031F3F7F7F3F1F16354484000000
000000C0FCFEFEFCF80C0A91
A1")
170 CALL CHAR(40,"00000E1115
0F61530F030103070F060E000070
88A87086CAF0C080C0E0F060
70")
180 CALL CHAR(47,"FFFF001C0C
7E1414")
190 CALL SCREEN(2):: CALL CO
LOR(0,2,2,1,11,2):: CALL COL
OR(14,8,2)
200 CALL CLEAR
210 CALL CHAR(33,"FFFFFFFF7E
3CDBFF"):: CALL CHAR(34,"000
0FF7E3C3C3C7E")
220 CALL CHAR(136,"FFFFC3E77
E3CFFFFC0FFC0C0FFC0C0FFC0C0F
FC0C0FFC0C0FFC0C0FFC0C0F
FC0"):: CALL CHAR(140,"03FF0
303FF0303FF0303FF0303FF0303F
F0303FF0303FF03")
240 CALL CHAR(132,"080F070F0
F0F0703010707 0FDFDC7C1E0080C
060F8F8E0C088FCFCC8E0E0E
0F0")
250 CALL CLEAR :: CALL DELSP
RITE(ALL):: VASE=0
260 CALL MAGNIFY(3)
270 RESTORE 9 :: CALL CHAR(1
43,"FFFF"):: CALL HCHAR(1,1,
143,32):: CALL HCHAR(24,
1,143,32)
280 FOR I=4 TO 20 STEP 4 ::
CALL HCHAR(I,1,136,32):: NEX
T I
290 CALL SPRITE(#1,128,2,168
,120)
300 FOR I=1 TO 12 :: READ A,
B :: CALL HCHAR(A,B,137):: C
ALL HCHAR(A+1,B,138):: C
ALL HCHAR(A+2,B,139)
310 CALL HCHAR(A,B+1,140)::
CALL HCHAR(A+1,B+1,141):: CA
LL HCHAR(A+2,B+1,142)::
NEXT I
320 RANDOMIZE :: FOR I=4 TO
20 STEP 8
330 P=INT(RND*31)+1 :: CALL
GCHAR(I-1,P,E1):: CALL GCHAR
(I+1,P,E2):: IF E1<>32 O
R E2<>32 THEN 330
340 CALL HCHAR(I,P,32):: NEX
T I
350 FOR I=4 TO 20 STEP 4
360 FOR J=1 TO 2
370 P=INT(RND*26)+2 :: CALL
GCHAR(I,P,E1):: CALL GCHAR(I
+1,P,E2):: CALL GCHAR(I-
1,P,E3):: IF E1=32 OR E2<>32
 OR E3<>32 THEN 370
380 CALL HCHAR(I-1,P,33):: C
ALL HCHAR(I-2,P,34):: NEXT J
 :: NEXT I
390 RUNDE=RUNDE+1 :: MOT=RUN
DE :: IF MOT>8 THEN MOT=8
400 CALL STAND(PUNKTE,RUNDE,
VASE):: IF RUNDE>5 THEN CALL
 SPRITE(#4,40,14,40,150,
0,-(MOT+5))
410 IF RUNDE>7 THEN CALL SPR
ITE(#5,40,12,8,59,0,MOT+5)
420 CALL SPRITE(#2,36,3,72,2
0,0,(5+MOT)/2)
430 IF RUNDE>1 THEN CALL SPR
ITE(#3,36,4,136,200,0,-(5+MO
T)/2)
440 IF RUNDE>3 THEN CALL SPR
ITE(#6,96,12,104,120,0,-MOT)
450 CALL COLOR(14,8,2,1,11,2
):: CALL COLOR(#1,9):: FOR I
=110 TO 310 STEP 10 :: C
ALL SOUND(-10,I,0,I+2,0,I+3,
0):: NEXT I :: CALL PONGOANZ
(PONGO)
460 CALL COINC(ALL,ST):: IF
ST=-1 THEN 690
470 CALL KEY(5,T,S):: CALL M
OTION(#1,0,0):: IF T=44 THEN
 570 ELSE IF T=46 THEN 5
80
480 IF T=65 THEN 490 ELSE IF
 T=90 THEN 530 ELSE 460
490 CALL POSITION(#1,R,C)::
R=R+2 :: R=R/8 :: C=C/8 :: I
F R<1 OR R>24 THEN 640
500 CALL GCHAR(R+2,C+.5,E1):
: CALL GCHAR(R+2,C+2,E2)
510 IF E1>135 OR E2>135 THEN
 CALL PATTERN(#1,100):: CALL
 MOTION(#1,-8,0):: CALL
SOUND(-1,392,0)
520 GOTO 460
530 CALL POSITION(#1,R,C)::
R=R-2 :: R=R/8 :: IF R>=20.5
 THEN 460 ELSE C=C/8
540 CALL GCHAR(R+4,C+.5,E1):
: CALL GCHAR(R+4,C+2,E2)
550 IF E1<>32 OR E2<>32 THEN
 CALL PATTERN(#1,100):: CALL
 MOTION(#1,8,0):: CALL S
OUND(-1,392,0)
560 GOTO 460
570 IF T=44 THEN CALL MOTION
(#1,0,-15):: CALL PATTERN(#1
,128):: CALL SOUND(-1,78
4,0):: GOTO 590
580 IF T=46 THEN CALL MOTION
(#1,0,15):: CALL PATTERN(#1,
132):: CALL SOUND(-1,784
,0):: GOTO 590
590 CALL POSITION(#1,X,Y)::
X=X+22 :: X=X/8 :: Y=Y/8 ::
CALL GCHAR(X,Y,E1):: CAL
L GCHAR(X,Y+1,E2):: IF E2=32
 OR E1=32 THEN 690
600 CALL GCHAR(X-1,Y,E1):: C
ALL GCHAR(X-1,Y+1,E2):: IF E
1=33 OR E2=33 THEN 610 E
LSE 460
610 CALL MOTION(#1,0,0):: CA
LL SOUND(-50,392,0,523,0,659
,0):: IF E1=33 THEN CALL
 VCHAR(X-2,Y,32,2):: PUNKTE=
PUNKTE+20+5*RUNDE :: VASE=VA
SE+1
620 IF E2=33 THEN CALL VCHAR
(X-2,Y+1,32,2):: PUNKTE=PUNK
TE+20+5*RUNDE :: VASE=VA
SE+1
630 CALL SOUND(-50,392,0,494
,0,587,0):: CALL STAND(PUNKT
E,RUNDE,VASE):: GOTO 460
640 IF VASE<7 THEN 690
650 PUNKTE=PUNKTE+500+10*RUN
DE :: CALL STAND(PUNKTE,RUND
E,VASE):: CALL MELODY
660 IF VASE=10 THEN BP=200+5
0*RUNDE :: CALL MELODY3(BP):
: PUNKTE=PUNKTE+BP :: CA
LL STAND(PUNKTE,RUNDE,VASE)
670 IF PUNKTE-BONUS*5000<500
0 THEN 250
680 PONGO=PONGO+1 :: BONUS=B
ONUS+1 :: CALL MELODY2 :: CA
LL PONGOANZ(PONGO):: GOT
O 250
690 CALL MOTION(#1,15,0):: C
ALL PATTERN(#1,104):: FOR I=
1100 TO 110 STEP -10 ::
CALL SOUND(-100,I,0,I+1,5,I-
1,5):: CALL POSITION(#1,Z,S)
:: IF Z>=160 THEN 710
700 NEXT I
710 CALL MOTION(#1,0,0):: CA
LL LOCATE(#1,167,S)
720 IF PUNKTE-BONUS*5000<500
0 THEN 740
730 PONGO=PONGO+1 :: BONUS=B
ONUS+1 :: CALL MELODY2
740 PONGO=PONGO-1 :: IF PONG
O=0 THEN 790
750 CALL PONGOANZ(PONGO):: C
ALL LOCATE(#1,167,120):: CAL
L PATTERN(#1,128)
760 CALL STAND(PUNKTE,RUNDE,
VASE)
770 ON ERROR 690
780 GOTO 460
790 CALL MELODY
800 DISPLAY AT(24,1):"     @
1983 VOLKER BECKER    "
810 FOR I=9 TO 11 :: DISPLAY
 AT(I,3)SIZE(24):" " :: NEXT
 I
820 DISPLAY AT(10,9)SIZE(9):
"GAME OVER" :: FOR I=1 TO 10
0 :: NEXT I
830 DISPLAY AT(10,9)SIZE(9):
" " :: FOR I=1 TO 80 :: NEXT
 I
840 CALL KEY(5,T,S):: IF S=0
 THEN 820
850 CALL DELSPRITE(ALL):: GO
TO 10
855 !@P+
860 SUB STAND(PUNKTE,RUNDE,V
ASE):: !@P-
870 DISPLAY AT(1,18)SIZE(6):
USING "######":PUNKTE
880 DISPLAY AT(1,6)SIZE(2):U
SING "##":RUNDE :: DISPLAY A
T(1,12)SIZE(2):USING "##
":VASE
885 !@P+
890 SUBEND
900 SUB PONGOANZ(PONGO)
905 GOTO 910 :: I :: !@P-
910 CALL HCHAR(24,3,143,28)
920 IF PONGO=1 THEN SUBEXIT
930 FOR I=1 TO PONGO-1 :: CA
LL SOUND(-20,220*2^(I/6),0):
: CALL HCHAR(24,2+I,47):
: NEXT I
935 !@P+
940 SUBEND
950 SUB MELODY
960 DATA 400,698,349,200,523
,262,200,30000,277,400,587,2
94,200,523,262,200,587,2
94,200,523,262,200,587,294
965 GOTO 970 :: I :: A :: B
:: C :: !@P-
970 DATA 200,659,330,200,698
,349,200,30000,262,200,30000
,175
980 RESTORE 960
990 FOR I=1 TO 12 :: READ A,
B,C :: CALL SOUND(A,B,0,C,0)
:: NEXT I
995 !@P+
1000 SUBEND
1010 SUB MELODY2
1020 DATA 200,262,200,330,45
0,392,200,330,450,392,450,78
4,200,659,450,784
1025 GOTO 1030 :: I :: A ::
B :: !@P-
1030 DISPLAY AT(1,12)SIZE(5)
:USING "#####":"BONUS"
1040 RESTORE 1010
1050 FOR I=1 TO 8 :: READ A,
B :: CALL SOUND(A,B,0):: NEX
T I
1060 FOR I=1 TO 150 :: NEXT
I
1070 CALL HCHAR(1,14,143,5)
1075 !@P+
1080 SUBEND
1090 SUB MELODY3(BP)
1100 DATA 262,220,247,196,22
0,175,196,165,175,147,165,13
1
1105 GOTO 1110 :: I :: A ::
!@P-
1110 RESTORE 1100
1120 DISPLAY AT(6,13)SIZE(4)
:USING "####":BP
1130 FOR I=1 TO 12 :: READ A
 :: CALL SOUND(-100,A,0,A+2,
0,A+3,0):: NEXT I
1140 DISPLAY AT(6,13)SIZE(4)
:" "
1145 !@P+
1150 SUBEND
10000 SUB EINLEITUNG
10005 GOTO 10010 :: T :: S :
: !@P-
10010 CALL CLEAR :: CALL DEL
SPRITE(ALL)
10020 DISPLAY AT(1,1):"ZIEL
DES SPIELES IST ES      PONGO
 AUF EINE DER BEIDEN    O
BEREN LEITERN ZU FUEHREN    U
ND DABEI SO VIELE VASEN     W
IE MOEGLICH ZU SAMMELN."
10030 DISPLAY AT(6,1):"PONGO
 KANN EINE DER LEITERN NUR D
ANN UNBESCHADET HINAUF-S
TEIGEN, WENN ER MINDESTENS 7
 VASEN AUF SEINEM WEG NACH O
BEN GESAMMELT HAT."
10040 DISPLAY AT(11,1):"BETR
ITT PONGO EINE DER ZWEI OBER
EN LEITERN OHNE MINDES-
TENS 7 VASEN ZU HABEN, SO
FAELLT ER VOM GERUEST."
10050 DISPLAY AT(15,1):"PONG
O FAELLT EBENFALLS VOM  GERU
EST, WENN ER AUF EINER
LEITER ZUR SEITE GEHT, UEBER
ODER IN DIE NAEHE EINES
LOCHES GEHT,DEN BILDSCHI
RM"
10060 DISPLAY AT(20,1):"AN D
EN SEITEN VERLAESST ODERVON
EINEM WESEN AUF DEM
GERUEST BERUEHRT WIRD."
10070 DISPLAY AT(24,0):"    W
EITER MIT E N T E R"
10080 CALL KEY(5,T,S):: IF T
=13 THEN 10090 ELSE 10080
10090 CALL CLEAR
10100 DISPLAY AT(1,1):"S T E
 U E R U N G :         -----
---------------"
10110 DISPLAY AT(4,1):"PONGO
 BEWEGT SICH MIT         ',' -
 NACH LINKS              '
.' - NACH RECHTS            '
A' - DIE LEITER HINAUF      '
Z' - DIE LEITER HINUNTER
."
10120 DISPLAY AT(10,1):"ZU S
PIELBEGINN STEHEN 5      ' PO
NGOS ' ZUR VERFUEGUNG."
10130 DISPLAY AT(12,1):"ALLE
 5000 PUNKTE GIBT ES     EINE
N ZUSAETZLICHEN PONGO."
10132 DISPLAY AT(16,1):"PUNK
TANZEIGE :                ----
-----------"
10134 DISPLAY AT(18,1):"LINK
S  - RUNDENZAHL           MITT
E  - ANZ.DER VASEN
RECHTS - PUNKTE"
10140 DISPLAY AT(24,1):"
WEITER MIT E N T E R"
10150 CALL KEY(5,T,S):: IF T
=13 THEN 10160 ELSE 10150
10160 CALL CLEAR
10170 DISPLAY AT(1,1):"PUNKT
WERTUNG :                -----
---------"
10180 DISPLAY AT(4,1):"FUER
JEDE GESAMMELTE VASE : 20 +
5 MAL RUNDENZAHL"
10190 DISPLAY AT(7,1):"FUER
DAS ERREICHEN EINER DEROBERE
N BEIDEN LEITERN MIT    M
INDESTENS 7 VASEN :         5
00 + 10 MAL RUNDENZAHL"
10200 DISPLAY AT(12,1):"WENN
 ES PONGO GELINGT IN     EINE
R RUNDE ALLE 10 VASEN ZU
SAMMELN :                    :
200 + 50 MAL RUNDENZAHL"
10210 DISPLAY AT(17,1):"----
-------------------------"
10220 DISPLAY AT(19,1):"NACH
 SPIELENDE IRGEND EINE  TAST
E DRUECKEN FUER EIN
NEUES SPIEL."
10225 DISPLAY AT(23,1):"** V
IEL SPASS BEIM SPIEL ** BITT
E DRUECKEN SIE E N T E R "
10230 CALL KEY(5,T,S):: IF T
=13 THEN 10240 ELSE 10230
10235 !@P+
10240 SUBEND
```

Scan: S. Höltgen

Eines der faszinierendsten und zugleich motivierendsten Erlebnisse mit dem Home-Computer ist das Entstehen von ersten selbstprogrammierten farbigen Mustern oder Figuren auf dem Bildschirm. Einige Bemerkungen hierzu sollen Ihnen den Weg dorthin erleichtern

Einfache Grafik selbst programmiert

Scan: S. Höllgen

So eigen wie im künstlerischen Bereich sind die verschiedenen Home-Computer-Modelle wohl auf keinem anderen Gebiet. Stirnrunzeln mit anschließender Fehlermeldung meldet der eine beim Versuch, ein Programm einzugeben, das beim anderen die bezauberndsten Ornamente entwarf: allgemeingültige Programme gibt es nicht.

Entscheidend ist die Auflösung

Wichtig für die Beurteilung der Grafikfähigkeiten sind: die Anzahl der Farben für Schrift, Hintergrund (und Rahmen), die Auflösungshöhe (sie bestimmt, wie fein gezeichnet werden kann) und das Vorhandensein der Möglichkeit, eigene Zeichen zu definieren.

Jeder Home-Computer teilt den Bildschirm in Zeilen und Spalten auf, so daß in jeder Position genau ein Zeichen Platz hat. Die Anzahl der Zeilen und der Spalten ist von Modell zu Modell verschieden. Bei allen Home-Computern kann man die einzelnen Positionen in beliebiger Reihenfolge mit irgendeinem Zeichen aus dem Zeichenvorrat des jeweiligen Modells beschreiben. Bei Home-Computern mit Farb-Fähigkeit ist es möglich, einem Zeichen in jeder Position eine beliebige der angebotenen Farben zuzuweisen.

Zufall bestimmt

Zur Erzeugung einfacher Farbenmuster genügt es schon, als Zeichen ein ausgefülltes Quadrat (das Gegenteil eines Leerzeichens) zu wählen und dieses zufällig einzufärben und an eine zufällige Position innerhalb des Zeilen-Spalten-Rasters zu bringen.

Es folgen unter Ausschöpfung einer möglichst großen Farbpalette und des ganzen Bildschirmraumes die Programme für sechs verschiedene Home-Computer.

Noch reizvoller sind symmetrische Muster. Sie kann man zum Beispiel erreichen, indem man den Zufall nur auf ein Viertel des Bildschirms beschränkt und die dort erzeugten Farbtupfer danach spiegelt.

Einen zusätzlichen Effekt erzielt man dabei, wenn man Spalte für Spalte weiterrückt und nur die Zeilenzahl und die Farbe zufällig erzeugt.

Aus drucktechnischen Gründen können wir leider die acht bzw. 16 Grafiksymbole für die Farben in der Zeile 10 der Commodore-Programme nicht abdrucken.

Zufällige Farbmuster für Spectrum

```
10 LET F=INT (RND*8)
20 LET Z=INT (RND*22)
30 LET S=INT (RND*32)
40 PRINT AT Z,S;INK F;
   "■"
50 GOTO 10
```

Zufällige Farbmuster für VC 20

```
10 P$="        "
20 D$="SQQQQQQQ
   QQQQQQQQQQ
   QQQQQQQQQ"
30 A=INT(RND(1)*8)+1
40 F$=MID$(P$,A,1)
50 Z=INT(22*RND(1))
60 S=INT(21*RND(1))
70 PRINT LEFT$(D$,
   Z+1);TAB (S);F$;"●"
80 GOTO 30
```

Zufällige Farbmuster für C 64

```
10 P$="                "
20 D$="SQQQQQQQ
   QQQQQQQQQQ
   QQQQQQQQQQ"
30 A=INT(RND(1)*16)+1
40 F$=MID$(P$,A,1)
50 Z=INT(24*RND(1))
60 S=INT(39*RND(1))
70 PRINT LEFT$(D$,
   Z+1);TAB (S);F$;"●"
80 GOTO 30
```

Zufällige Farbmuster für TI-99/4A

```
10 CALL CHAR(33,
   "FFFFFFFFFFFFFFFF")
20 F=INT(RND*16)+1
30 Z=INT(RND*24)+1
40 S=INT(RND*32)+1
50 CALL COLOR(1,F,1)
60 CALL
   HCHAR(Z,S,33,1)
70 GOTO 20
```

Zufällige Farbmuster für Dragon 32

```
10 F=RND(8)-1
20 P=RND(511)-1
30 PRINT@P,CHR$(143
   +F*16);
40 GOTO 10
```

Zufällige Farbmuster für Atari

```
10 GRAPHICS 3+16
20 F=INT (RND(1)*4)
30 Z=INT(RND(1)*24)
40 S=INT(RND(1)*40)
50 COLOR F
60 PLOT S,Z
70 GOTO 20
```

Symmetrische Farbmuster für Spectrum

```
10 FOR S=0 TO 15
20 LET Z=INT(RND*11)
30 LET F=INT(RND*8)
40 INK F
50 PRINT AT Z,S;"■"
60 PRINT AT Z, 31-S;"■"
70 PRINT AT 21-Z,S;"■"
80 PRINT AT 21-Z,
   31-S;"■"
90 NEXT S
100 GOTO 10
```

Symmetrische Farbmuster für VC 20

```
10 P$="        "
20 D$="SQQQQQQQ
   QQQQQQQQQQ
   QQQQQQ"
30 FOR S=0 TO 10
40 Z=INT(RND(1)*11)
50 A=INT(RND(1)*8)+1
60 F$=MID$(P$,A,1)
70 PRINT F$
80 PRINT LEFT$
   (D$,Z+1);TAB(S);"●"
90 PRINT LEFT$(D$,Z+1);
   TAB(20-S);"●"
100 PRINT LEFT$(D$,21-
   Z);TAB(S);"●"
110 PRINT LEFT$(D$,21-
   Z);TAB(20-S);"●"
120 NEXT S
130 GOTO 30
```

Symmetrische Farbmuster für C 64

```
10 P$="                "
20 D$="SQQQQQQQ
   QQQQQQQQQQ
   QQQQQQQQQ"
30 FOR S=0 TO 19
40 Z=INT(RND(1)*12)
50 A=INT(RND(1)*16)+1
60 F$=MID$(P$,A,1)
70 PRINT F$
80 PRINT LEFT$(D$,
   Z+1);TAB(S);"●"
90 PRINT LEFT$(D$,
   Z+1);TAB(38-S);"●"
100 PRINT LEFT$
   (D$, 23-Z); TAB(S);"●"
110 PRINT LEFT$(D$,23-
   Z);TAB(38-S);"●"
120 NEXT S
130 GOTO 30
```

Symmetrische Farbmuster für TI-99/4A

```
10 CALL CHAR(33,
   "FFFFFFFFFFFFFFFF")
20 FOR S=1 TO 16
30 Z=INT(RND*12)+1
40 F=INT(RND*16)+1
50 CALL COLOR(1,F,1)
60 CALL HCHAR
   (Z,S,33,1)
70 CALL HCHAR
   (Z,33-S,33,1)
80 CALL HCHAR
   (25-Z,S,33,1)
90 CALL HCHAR
   (25-Z,33-S,33,1)
100 NEXT S
110 GOTO 20
```

Symmetrisches Farbmuster für Dragon 32

```
10 FOR S=0 TO 15
20 Z=RND(8)-1
30 F=RND(8)-1
40 PRINT@Z*32+S,
   CHR$(143+F*16);
50 PRINT@Z* 32+30-S,
   CHR$(143+F*16);
60 PRINT@(15-Z)*32+S,
   CHR$(143+F*16);
70 PRINT@(15-Z)*32+30-
   S,CHR$(143+F*16);
80 NEXT S
90 GOTO 10
```

Symmetrisches Farbmuster für Atari

```
10 GRAPHICS 3+16
20 FOR S=0 TO 19
30 Z=INT(RND(1)*12)
40 F=INT(RND(1)*4)
50 COLOR F
60 PLOT S,Z
70 PLOT S,23-Z
80 PLOT 39-S,Z
90 PLOT 39-S,23-Z
100 NEXT S
110 GOTO 20
```

Scan: S. Höltgen

Fregatte

Man kann auf dem TI 99/4A mit Extended-BASIC an einer äußerst realistisch aufgebauten Seeschlacht teilnehmen. Bei diesem Programm aus der HC-Diplom-Aktion heißt es, Augen und Ohren offenhalten, um die gegnerischen Schiffe ins Fadenkreuz zu bekommen

Man befindet sich auf der Brücke eines Schiffes unterhalb des Blickfeldes, in dem ein nach oben und unten bewegliches Fadenkreuz angebracht ist und alle wichtigen Daten sind. Durch die entsprechenden Steuertasten kann man das Schiff links- oder rechtsherum mit verschiedenen Geschwindigkeiten um seine Achse drehen.

Optisch wurde das so realisiert, daß sich alle Objekte im Blickfeld in die entgegengesetzte Richtung bewegen. Der einzige sichere Bezugspunkt hierbei ist die Sonne über dem Horizont, da sich die Schiffe im Blickfeld ja ohnehin bewegen können.

Die Gegner sind unberechenbar

Die gegnerischen Schiffe nähern sich dem eigenen auf unregelmäßigen Zickzackbahnen. Je näher sie kommen, desto größer werden sie. Dadurch ergibt sich ein sehr realistisch wirkendes Blickfeld.

Wenn ein gegnerisches Schiff eine bestimmte Entfernung erreicht hat, bewegt es sich nur noch nach links oder rechts und gibt in unregelmäßigen Abständen Schüsse auf das eigene Schiff ab; nicht jeder trifft, aber die Anzahl der Schüsse und der Anteil der Treffer nehmen mit jeder Runde zu.

Die Verteidigung

Nun zur eigenen Kanone: Man hat pro Runde 33 Schuß. Das bedeutet, daß man mit seiner Munition haushalten muß. Äußerst realistisch und auch schwer ist das Spiel dadurch, daß die Flugdauer der Ge-

```
100 CALL CLEAR :: CALL SCREE
N(2):: CALL CHARSET
110 CALL CHAR(35,"3C4299A1A1
99423C")
120 PRINT "          FREGATT
E"
130 PRINT :;:;:;:;:;:;:
140 PRINT "      [1] SPIEL ST
ARTEN"
150 PRINT "      [2] ANLEITUN
G"
160 PRINT "      [3] ENDE"
170 PRINT :;:;:;:;:;:
180 PRINT "   # BY ANDREAS E
HLERDING"
190 GOSUB 1660
200 IF KE<49 OR KE>51 THEN 1
20
210 ON KE-48 GOTO 440,220,17
60
220 PRINT "HIER GILT ES EIN
FURCHTBARESSEEGEMETZEL ZU UE
BERSTEHEN :"
230 PRINT " DU BEFINDEST DIC
H IN EINER FREGATTE,DIE VON
ZAHLREICHENSCHIFFEN EING
EKESSELT IST.   SCHAFFST DU
ES,DIR DEINE"
240 PRINT "GEGNER VOM HALS Z
U HALTEN ?  WENN DEINE MUNIT
ION VER-   BRAUCHT IST U
ND DU MEHR ALS 10000 PUNKTE
ERREICHT HAST,"
250 PRINT "KOMMT EIN VERSORG
UNGSSCHIFF,DASS DEINE MUNITI
ONSDEPOTS  WIEDER AUFFUE
LLT. DANACH"
260 PRINT "BEGINNT SOFORT EI
N NEUES GE-FECHT.DIESMAL MIT
 GEFAEHR-  LICHREN GEGNE
RN..."
270 PRINT " DEINE STEUERMOEG
LICHKEITEN FINDEST DU AUF DE
R NAECHSTENSEITE DER ANL
EITUNG :"
280 GOSUB 1660
290 PRINT "[,] SCHIFF N. LIN
KS DREHEN  [.] SCHIFF N. REC
HTS DREHEN"
300 PRINT "[;] SCHIFF STOPPE
N"
310 PRINT "[A] FADENKREUZ AU
FWAERTS    [Z] FADENKREUZ AB
WAERTS     [X] FEUERN ,B
IS ZU DREI       SCHUSS IN EIN
ER SALVE,DANN"
320 PRINT "MUSST DU ERST WAR
TEN BIS    MINDESTENS EIN SC
HUSS NACH- GELADEN IST.
IN DER ANZEIGE UNTER DEM FAD
ENKREUZ SIND"
330 PRINT "ALLE WICHTIGEN IN
FORMATIONENZUR KANONE UND DE
INE PUNKT- ZAHL ZU FINDE
N. LINKS,NEBEN DEM  KREUZ ST
EHT DIE ANZAHL"
340 PRINT "DER TREFFER DIE D
U ERHALTEN HAST (MAX.9). DAN
EBEN BEFIN-DET SICH DIE
ANZAHL DER VON DIR VERSENKTE
N SCHIFFE."
350 PRINT "RECHTS STEHT DIE
ANZAHL     DEINER VERSORGUNS
SCHIFFE."
360 GOSUB 1660
370 PRINT "BEIM SCHIESSEN SO
LLTEST DU  FOLGENDES BEACHTE
N :          IM SINNE EINE
R MOEGLICHST   NATURGETREUEN
 REPRODUKTION,"
380 PRINT "SCHIESST DEIN SCH
IFF NICHT  ENTLANG EINER BAH
N (WIE DIESBEI VIELEN SP
IELEN DER FALL IST) SONDERN
AUF EINEN PUNKT"
390 PRINT ",DEN DU DURCH DIE
 STELLUNG  DES FADENKREUZES
BESTIMMST. DAS HEISST,DA
SS SCHIFFE DIE SICH ZWAR AUF
 DER BAHN,ABER"
400 PRINT "NICHT IN DER PASS
ENDEN ENT- FERNUNG BEFINDEN,
UNBEHELLIGTBLEIBEN. AUSS
ERDEM STEIGT   MIT DER ENTFE
RNUNG DES ZIELS"
410 PRINT "AUCH DIE DAUER DE
S GESCHOSS-FLUGES (WIRKLICHK
EITSNAEHE!),WESHALB NATU
ERLICH DIE BE- WEGUNG DES ZI
ELES BERUECK-  SICHTIGT WERD
EN MUSS."
420 GOSUB 1660
430 GOTO 120
440 PRINT "SPIELSTUFE WAEHLE
N : [1]-[5]"
450 PRINT :;:;:;:;:;:
460 GOSUB 1660
470 CALL KEY(0,KE,ST):: IF S
T=0 THEN 470
480 R=KE-48 :: IF R<1 OR R>5
 THEN 440
490 FV=2 :: PU,ES=0
500 DATA 3,16,4,15,5,4,6,11,
8,10
510 DIM S(4,1),EX(4),VH(3),P
V(3),TAS(3,2)
520 CALL CLEAR :: CALL SCREE
N(2):: CALL MAGNIFY(4)
530 CALL CHAR(96,"0000000000
000FFF00000000FFFFFF00000F
FFFFFFFFF0FFFFFFFFFFFF
FF")
540 CALL CHAR(100,"F0FFFFFFF
FFFFFFF0000F0FFFFFFFFFF00000
000F0FFFFFF000000000000F
0FF")
550 CALL COLOR(9,2,5)
560 CALL CHAR(104,"000000000
0000FFF000000000FFFFFF00000
FFFFFFFFFFF0FFFFFFFFFFFF
FFF")
570 CALL CHAR(108,"F0FFFFFFF
FFFFFFF0000F0FFFFFFFFFF00000
000F0FFFFFF000000000000F
0FF")
580 CALL COLOR(10,14,2)
590 CALL CHAR(112,"")
600 CALL CHAR(114,"0103070F1
F3F7FFF80C0E0F0F8FCFEFF")
610 CALL CHAR(116,"FF8181818
18180FCFC808181818181FFFF818
1818181013F3F01818181818
1FF")
620 CALL COLOR(11,2,5)
630 CALL CHAR(120,"FFFFFFFFF
FFFFFFF")
640 CALL CHAR(121,"")
650 CALL CHAR(122,"0103070F1
F3F7FFF80C0E0F0F8FCFEFF")
660 CALL COLOR(12,14,2)
670 CALL HCHAR(4,1,120,320):
: CALL HCHAR(13,1,112,384)
680 CALL HCHAR(21,1,121,128)
:: CALL HCHAR(1,1,121,96)
690 D$="0401yyyyyyyyhij{yyyy
yyyyzmnoyyyyyyyy"
700 GOSUB 1480
710 D$="0501yyyyhijkxxxx{yyy
yyyzxxxxlmnoyyyy"
720 GOSUB 1480
730 D$="0601hijkxxxxxxxxx{yy
yyzxxxxxxxxxlmno"
740 GOSUB 1480
750 D$="2001yyyydefgppprYYYY
yyyysppp`abcyyyy"
760 GOSUB 1480
770 D$="1901defgppppppppryyy
yyyspppppppp`abc"
780 GOSUB 1480
790 CALL CHAR(33,"0010144822
0C3C0000187E7E181818000000060
7E30780000"):: CALL CHAR
(58,"00010103073F1A0F0000206
0F4F4BEFE")
800 RESTORE 500
810 FOR X=1 TO R :: READ FH,
FS :: NEXT X
820 FOR X=1 TO 4 :: CALL COL
OR(X,2,FH):: NEXT X
830 DISPLAY AT(20,13)SIZE(4)
:"#  3" :: DISPLAY AT(21,13)
SIZE(4):"# 33" :: DISPLA
Y AT(23,12)SIZE(6):PU
840 DISPLAY AT(24,2)SIZE(3):
CHR$(34)&" 0" :: DISPLAY AT(
24,7)SIZE(4):"! 0" :: DI
SPLAY AT(24,19)SIZE(6):":;
"&STR$(ES)
850 SG=1+R/5 :: SR=33 :: VHS
,TR,TG,AS=0 :: RANDOMIZE ::
FOR X=1 TO 3 :: TAS(X,0)
,PV(X),VH(X)=0 :: NEXT X
860 CALL SPRITE(#1,116,2,91,
113):: FK=91
870 CALL CHAR(84,"030F1F3F7F
7FFFFFFFFF7F7F3F1F0F03C0F0F8
FCFEFEFFFFFFFFFEFEFCF8F0
C0")
880 IF R=1 THEN CALL SPRITE(
#28,84,12,49,RND*200+1)
890 CALL CHAR(128,"010103030
7070F0F1F1F3F3F7F7FFFFF8080C
0C0E0E0F0F0F8F8FCFCFEFEF
FFF")
900 CALL CHAR(132,"000000000
0010307000000000000000000000
0000080C0E000000000000000
000")
910 CALL CHAR(136,"000001050
5030B07000000000000000000000
0A0C080D0E000000000000000
000")
920 CALL CHAR(140,"0002010F0
70F7F1F380000000000000000004
040D0F0F8E070000000000000
000")
930 CALL CHAR(76,"0105130F27
3F4F1FFF4000000000000000808090
F0E0FCC0F03E10")
940 CALL CHAR(80,"0105130F27
3F4F1FFF4F087C10100000080C090
F0C4FCC0F8FEF0C0F0")
950 EX(0)=132 :: EX(1)=136 :
: EX(2)=140 :: EX(3)=76 :: E
X(4)=80
960 CALL CHAR(36,"0000000000
0000010100000000000000000000
000000008080")
```

schosse von der Entfernung abhängig ist, auf die das Fadenkreuz beim Schuß eingestellt war. Übrigens braucht man sich auch keine Sorgen zu machen, wenn man aus der Drehung schießt oder sich nach dem Abschuß in eine andere Richtung dreht: Der Schuß trifft dort auf, wo das Fadenkreuz im Augenblick des Abfeuerns stand.

```
970 CALL CHAR(40,"0000000000
000003070300000000000000000
000000080C0E0C0")
980 CALL CHAR(44,"0000000000
000103070300000000000000000
00000000080E0C0")
990 CALL CHAR(60,"0000000000
0206071F0F0000000000000000000
00000040C0E0F0")
1000 CALL CHAR(64,"000000000
0000203070F00000000000000000
000004060E0F8F0")
1010 CALL CHAR(68,"000000010
30F0FFF7F3F0000000000000000
000009090F8FEFE")
1020 CALL CHAR(72,"000000000
009091F7F7F00000000000000000
080C0F0F0FFFEFC")
1030 CALL CHAR(92,"040705010
7040703071F0F07030300020E0A
080E020E0C0E0F8F0E0C0C0"
)
1040 S(0,0),S(0,1)=36 :: S(1
,0)=44 :: S(1,1)=40 :: S(2,0
)=60 :: S(2,1)=64 :: S(3
,0)=68 :: S(3,1)=72 :: S(4,0
),S(4,1)=92
1050 FOR X=21 TO 23 :: CALL
SPRITE(#X,S(0,0),FS,81,1+RND
*150):: NEXT X
1060 FOR X=1 TO 3 :: TAS(X,2
)=X+4 :: NEXT X
1070 CALL KEY(1,A,B):: CALL
KEY(2,C,D):: IF B=0 THEN 113
0
1080 IF A=0 AND AS<3 THEN 15
00
1090 FK=SGN(A-2)*5+FK
1100 IF FK>101 THEN FK=101
1110 IF FK<81 THEN FK=81
1120 CALL LOCATE(#1,FK,113)
1130 IF D=0 THEN 1150 ELSE I
F C=14 THEN VHS=VHS+2 ELSE I
F C=13 THEN VHS=VHS-2 EL
SE VHS=0
1140 IF ABS(VHS)>10 THEN VHS
=SGN(VHS)*10
1150 FOR X=1 TO 3 :: CALL MO
TION(#20+X,0,VH(X)+VHS,#4+X,
0,VHS):: NEXT X
1160 CALL MOTION(#28,0,VHS)
1170 IF AS=0 THEN 1200
1180 FOR X=1 TO AS :: TAS(X,
0)=TAS(X,0)-1 :: IF TAS(X,0)
=0 THEN 1540
1190 NEXT X
1200 CALL SOUND(-4250,-4,10-
ABS(VHS),110,30,110,30,230+A
BS(VHS)*15,30):: X=INT(S
QR(RND*9))+1 :: PVM=INT(RND*
SG):: IF PVM=0 THEN 1240
1210 IF PV(X)=4 THEN 1260 EL
SE PV(X)=PV(X)+1
1220 CALL POSITION(#20+X,V,H
):: IF H=0 THEN H=1
1230 CALL LOCATE(#20+X,PV(X)
*5+81,H)
1240 VH(X)=VH(X)+(RND-.5)*(P
V(X)+1):: CALL PATTERN(#20+X
,S(PV(X),SGN(1+SGN(VH(X)
))))
1250 GOTO 1310
1260 CALL POSITION(#20+X,V,H
):: IF H=0 THEN H=1
1270 CALL SPRITE(#4,132,16,1
01,H):: CALL SOUND(-200,-5,0
):: CALL COLOR(#4,12)::
CALL SOUND(-200,-6,3):: CALL
 COLOR(#4,16):: CALL SOUND(-
400,-7,6)
1280 CALL DELSPRITE(#4):: Y=
INT(RND*SG):: IF Y=0 THEN 13
10
1290 TG=TG+1 :: DISPLAY AT(2
4,3)SIZE(2):TG :: FOR Y=1 TO
 4 :: CALL SOUND(-100,-8
,Y*3,220,30,220,30,1000-Y*20
0,30):: CALL COLOR(Y,9,2)::
NEXT Y
1300 FOR Y=1 TO 4 :: CALL CO
LOR(Y,FV,FH):: NEXT Y :: CAL
L SOUND(-400,-8,15,220,3
0,220,30,200,30):: IF TG=10
THEN 1420
1310 IF SR THEN 1070
1320 IF ES=0 THEN 1420
1330 CALL SPRITE(#4,S(3,0),2
,96,256,0,-8):: CALL MOTION(
#28,0,0)
1340 FOR X=1 TO 3 :: CALL PA
TTERN(#20+X,EX(PV(X))):: CAL
L COLOR(#20+X,16):: CALL
 SOUND(-200,-8,X*2,220,30,22
0,30,800+200*X,30)
1350 CALL COLOR(#20+X,12)::
CALL SOUND(-400,-8,2*X+2,220
,30,220,30,500+300*X,30)
:: CALL COLOR(#20+X,16):: CA
LL DELSPRITE(#20+X,#4+X):: N
EXT X
1360 CALL HCHAR(24,4,32):: C
ALL HCHAR(24,9,32):: CALL HC
HAR(24,21,32,2):: CALL V
CHAR(20,15,32,2)
1370 RESTORE 1710 :: READ N
:: FOR X=1 TO N :: READ DU,F
Q :: CALL SOUND(DU,FQ,3,
FQ+1,5,FQ*1.5,7):: NEXT X
1380 FOR X=1 TO 300 :: NEXT
X :: CALL DELSPRITE(#4):: R=
R+1 :: ES=ES-1 :: IF R<6
 THEN 790
1390 CALL DELSPRITE(#1,#28):
: CALL CHARSET :: FOR X=1 TO
 8 :: CALL COLOR(X,2,5):
: NEXT X :: CALL COLOR(9,2,5
,11,2,5,10,14,2,12,14,2)
1400 DISPLAY AT(14,3):"DAS W
AR DIE HOELLE !!!" :: DISPLA
Y AT(16,2):"ABER DU HAST
 ES GESCHAFFT." :: DISPLAY A
T(17,2):"**********************
*******"
1410 RESTORE 1730 :: READ Y
:: FOR X=1 TO Y :: READ DU,F
Q :: CALL SOUND(DU,FQ,2,
FQ+1,4,FQ*1.5,6):: NEXT X ::
 FOR X=1 TO 1000 :: NEXT X :
: CALL CLEAR :: GOTO 176
0
1420 CALL SCREEN(2):: CALL C
LEAR :: CALL DELSPRITE(ALL):
: CALL CHARSET :: FOR X=
1 TO 8 :: CALL COLOR(X,16,2)
:: NEXT X
1430 DISPLAY AT(2,2):"HERZLI
CHEN GLUECKWUNSCH !" :: FOR
X=1 TO 1000 :: NEXT X ::
 DISPLAY AT(6,1):"ZU DEINEM
ENTSCHLUSS,DEINE"
1440 DISPLAY AT(8,3):"MARINE
KARIERE VOR DEINEM" :: DISPL
```


DAS COMPUTER PROGRAMM

Der Alphacom 32

Thermopapier Printer mit eingebautem Interface für den Sinclair ZX81 und ZX Spectrum. Ein ungewöhnlich robuster Rollenpapier-Printer, kompakt und handlich, extrem problemlos in der Anwendung, unermüdlich funktionstüchtig dank eingebauter Ventilation. Energieversorgung durch separaten Power-Adaptor. Einfacher Steckanschluß an Ihren Computer.

Und das bietet der Alphacom 32: Durch einfachen Steckeranschluß sofort betriebsbereit. Eingebautes Interface für den Sinclair ZX81 und ZXSpectrum, mit allen Befehlen voll Sinclair-kompatibel. 32 Zeichen pro Zeile. Ausdruck aller Grafikzeichen. Hochauflösende Grafik. Sehr klarer Ausdruck auf Thermopapier. Druckt 2 Zeilen pro Sekunde.
Lieferumfang: Drucker mit eingebautem Interface, Steckanschlüsse, eine Rolle Thermopapier, Power-Adaptor und ausführliche Beschreibung.
Nr. 106,

DM 298,–

COMPUTER ACCESSOIRES INT'L
Der Computer-Ausstatter.

Hier wird bestellt:

☐ per Vorausscheck
☐ per Nachnahme (zuzügl. Nachnahmegeb.)

Stück	Alphacom 32 Thermopapier Printer Nr. 106	Einzel-Preis 298,–	Gesamt-Preis

Name
Straße
PLZ/Ort
Datum
Unterschrift

Bei Bestellungen unter DM 250,– zuzügl. Versandspesen.

COMPUTER ACCESSOIRES INT'L GMBH
Jägerweg 10 – 8012 Ottobrunn HC

Da es bis zu sieben Sekunden dauern kann, bis das Geschoß sein Ziel erreicht, wurde das Programm so gestaltet, daß es bis zu drei Schüsse zugleich verwalten kann. Aus dem gleichen Grund muß man natürlich die Eigenbewegung, besonders der weiter entfernten Schiffe, beim Schießen mit einkalkulieren.

Entfernte Schiffe zählen mehr

Dies könnte natürlich leicht zu der Versuchung führen, nur auf die nahe gelegenen Schiffe zu schießen. Gegen diese Taktik sprechen jedoch zwei Programmeigenschaften:

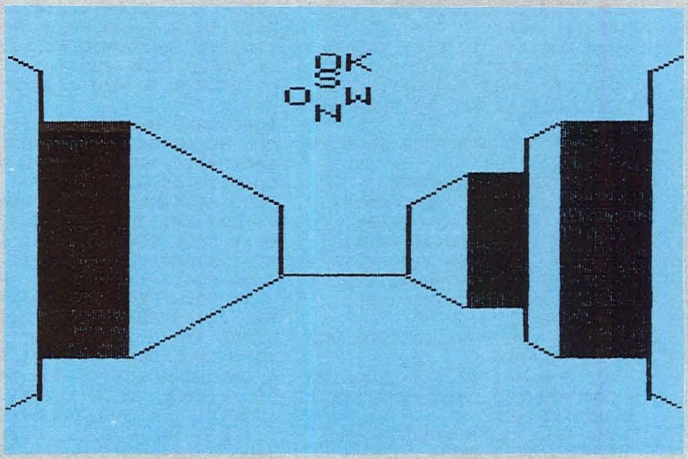
1. in der Nähe befindliche Schiffe bewegen sich mit einer scheinbar höheren Geschwindigkeit – wie dies ja auch in der Realität der Fall wäre

2. die Punktzahl für ein getroffenes Schiff steigt mit seiner Entfernung

Jedes der drei angreifenden Schiffe hat in seinen Bewegungen einen individuellen Charakter. Da es dem Programmierer bei der Entwicklung nicht so sehr auf Schnelligkeit, sondern auf eine raffinierte Spieltaktik ankam, wird das Spiel so schnell nicht langweilig. Und nun viel Spaß beim Spielen. *Andreas Ehlerding*

```
AY AT(10,2):"PANTOFFELKI
NO ZU BEGINNEN." :: DISPLAY
AT(12,1):"SCHONE BITTE AUCH
WEITERHIN"
1450 DISPLAY AT(14,1):"DEINE
 SEEFAHRENDEN KAMERADEN" ::
DISPLAY AT(16,3):"UND TR
AINIER BESSER NOCH"
1460 RESTORE 1750 :: FOR X=1
 TO R :: READ A$ :: NEXT X :
: DISPLAY AT(18,INT((28-
LEN(A$))/2)):A$ :: DISPLAY A
T(20,1):"BEVOR DU DIE HAIE F
UETTERST!"
1470 A$=STR$(PU)&" PUNKTE" :
: DISPLAY AT(22,INT((28-LEN(
A$))/2)):A$ :: FOR X=1 T
O 4000 :: NEXT X :: CALL CLE
AR :: GOTO 100
1480 VD=VAL(SEG$(D$,1,2))::
HD=VAL(SEG$(D$,3,2))
1490 FOR D=HD TO HD+LEN(D$)-
5 :: CALL VCHAR(VD,D,ASC(SEG
$(D$,D-HD+5,1))):: NEXT
D :: RETURN
1500 CALL SPRITE(#4,128,16,1
13,113,#TAS(AS+1,2),36,2,1,1
13,0,VHS):: AS=AS+1 :: C
ALL COLOR(#4,12):: CALL COLO
R(#4,16):: CALL COLOR(#4,11)
1510 CALL DELSPRITE(#4):: CA
LL SOUND(-200,-8,0,110,30,11
0,30,9000,30):: SR=SR-1
1520 TAS(AS,0)=8-INT((FK-80)
/3):: TAS(AS,1)=FK :: CALL S
OUND(-300,-8,8,110,30,11
0,30,9045,30)
1530 DISPLAY AT(20,15)SIZE(2
):3-AS :: DISPLAY AT(21,14)S
IZE(3):SR :: GOTO 1130
1540 AS=AS-1 :: T=TAS(X,2)
1550 CALL POSITION(#T,V,H)::
 IF H=0 THEN H=1
1560 CALL LOCATE(#T,TAS(X,1)
,H):: CALL PATTERN(#T,EX((TA
S(X,1)-81)/5)):: CALL CO
LOR(#T,16):: CALL COLOR(#T,1
2)
1570 FOR Y=21 TO 23 :: CALL
SOUND(-200,-8,(Y-21)*3,220,3
0,220,30,(24-Y)*50+TAS(X
,1)*30,30):: CALL COLOR(#T,1
6):: CALL COINC(#T,#Y,8,CO)
1580 IF CO THEN 1630
1590 CALL COLOR(#T,12):: NEX
T Y
1600 CALL DELSPRITE(#T)
1610 FOR Y=X TO 2 :: TAS(Y,0
)=TAS(Y+1,0):: TAS(Y,1)=TAS(
Y+1,1):: TAS(Y,2)=TAS(Y+
1,2):: NEXT Y :: TAS(3,2)=T
1620 DISPLAY AT(20,15)SIZE(2
):3-AS :: GOTO 1190
1630 CALL DELSPRITE(#Y):: TR
=TR+1 :: PU=PU+80*(106-TAS(X
,1)):: DISPLAY AT(23,12)
SIZE(6):PU :: IF PU>(R+ES)*1
0000 THEN ES=ES+1 :: DISPLAY
 AT(24,22)SIZE(2):ES
1640 CALL SPRITE(#Y,36,FS,81
,RND*200+1):: VH(Y-20),PV(Y-
20)=0
1650 CALL SOUND(-100,-8,0,11
0,30,110,30,20000,30):: DISP
LAY AT(24,8)SIZE(3):TR :
: GOTO 1590
1660 PRINT
1670 PRINT "   BITTE TASTE D
RUECKEN !"
1680 CALL SCREEN(8)
1690 CALL KEY(0,KE,ST):: IF
ST=0 THEN 1690
1700 CALL SCREEN(2):: CALL C
LEAR :: RETURN
1710 DATA 13,300,147,300,165
,150,147,150,165,450,175,300
,196,150,175,150,165,300
,175,150,220,150,247,300,220
1720 DATA 450,110
1730 DATA 21,300,262,150,294
,150,330,150,349,150,494,300
,440,300,524,300,440,150
,494,150,588,150,494
1740 DATA 150,588,300,494,15
0,524,150,494,150,440,300,39
2,150,440,150,494,150,39
2,300,330
1750 DATA EINIGE JAHRE,2 MON
ATE,EINE WOCHE,EIN PAAR STUN
DEN,EIN GANZ BISCHEN
1760 END
```

Falltür



Mit dem Auftrag, dreizehn Kristalle zu finden, befindet man sich in einem dreidimensionalen Labyrinth. Ein aufregendes Programm aus der HC-Diplom-Aktion für den VC 20 mit 16K-Erweiterung

Das Programm startet mit der einleiten¿en Erklärung, und dann entsteht auf dem Bildschirm nichts weiter als eine Wand. Grund: Man beginnt das Spiel am Eingang mit Blick nach Norden, und dort ist nun 'mal zunächst nichts besonderes zu sehen. Dreht man sich jetzt nach Osten, sieht man einen dargestellten Gang und eine Abzweigung nach rechts. Immer wenn man in einen Nebengang abbiegen will, muß man ihn neben sich haben, auf dem Bildschirm darf er also nicht mehr zu sehen sein.

Nun kann man sich seinem eigentlichen Auftrag widmen, nämlich die dreizehn im Labyrinth versteckten Kristalle zu finden, wobei vielerlei Gefahren in Form von Spinnen oder Falltüren, aber auch Hilfen wie Schwerter oder Mäntel auftauchen können.

Wenn am oberen Rand des Bildschirms „o. k." erscheint, kann man ab sofort wieder seinen nächsten Zug eingeben. Dies sollte schnell geschehen, da man jederzeit von einer Spinne angegriffen werden kann oder sich eine Falltür öffnen kann, auf der man gerade steht.

Gesteuert wird mit den Tasten n (nach Norden), s (nach Süden), w (nach Westen) und o (nach Osten).

Es besteht die Möglichkeit, den Spielstand mit a abzuspeichern und mit l wieder zu laden. Falls ein Drucker vorhanden ist, kann ferner noch mit der Taste „*" eine Bildschirm-Hardcopy erstellt werden.

Vor dem Laden zu beachten

Bevor man das Programm lädt, muß zunächst im Direktmodus POKE44,28:POKE7168,0:NEW eingegeben werden, um den Bereich, in dem das Labyrinth abgelegt ist, vor dem Überschreiben zu schützen.

Uwe Hauck

Scan: S. Höltgen

Ein Labyrinth entsteht...

Mit einem einfachen Plan lassen sich Labyrinthe entwerfen. Setzt man ihn in ein Computer-Programm um und lenkt den Zufall in die richtigen Bahnen, so sind dem Einfallsreichtum des Home-Computers keine Grenzen gesetzt

Man nehme ein Blatt Rechenpapier, einen Bleistift und male ein Rechteck auf. Damit wären schon einmal die Außenmauern des Irrgartens errichtet, die später ein Eingangstor und ein Ausgangstor erhalten.

Die zündende Idee

Zum Einziehen der Innenmauern gehe man folgendermaßen vor: Man beginne irgendwo auf einem Kreuzungspunkt zweier Linien, der noch von keiner Mauer berührt wird, ziehe entlang der Linien Mauerstück an Mauerstück und setze den Bleistift erst wieder ab, wenn man auf eine bereits bestehende Mauer trifft.

Bei den Aufbauarbeiten sind akkurat gerade Mauern ebenso erlaubt wie chinesische; nur von den Linien darf man nicht abweichen. Auf diese Weise ziehe man eine Wand nach der anderen ein, indem man jedesmal zunächst nach einem freien Kreuzungspunkt ohne Wandberührung sucht und dann von dort aus einen torkelnden Maurer auf Reisen schickt, bis er vor eine bestehende Wand läuft. Was eigentlich selbstverständlich erscheint, bei der späteren Formulierung des Computer-Programms jedoch berücksichtigt werden muß: Für den vagabundierenden Maurer gibt es kein Zurück mehr in die Richtung, aus der er gerade kam. Außerdem wollen wir ihn vor dem Mißgeschick bewahren, daß er sich versehentlich einmauert und in seiner Isolation elendiglich verhungern muß: Eine Wand darf also niemals beendet werden, indem sie auf sich selbst trifft. Diese letzte Bedingung verhindert, daß im Labyrinth Gebiete entstehen, zu denen es hinterher keinen Zugang gibt.

Von der Idee zum Programm

Nach den genannten Bauvorschriften entstandene Labyrinthe haben alle die Eigenschaft, daß es von jeder Stelle des Irrgartens zu jeder anderen Stelle genau einen Weg gibt, keinen mehr und keinen weniger. Somit ist es auch egal, wohin man den Eingang und den Ausgang verlegt: Es gibt auf jeden Fall immer genau einen Weg.

Was bisher beschrieben wurde, nennt man auch einen Algorithmus. Er muß jetzt in die einzelnen Schritte eines Programmes umgesetzt werden, um anschließend auf dem Home-Computer abzulaufen. Wie nach dem Beschriebenen klar ist, muß das Programm stets wissen, wo sich schon eine Wand befindet; diese Stellen stehen ja zum Beispiel für einen Neubeginn einer Wand nicht mehr zur Verfügung.

Wie stets bei Problemen, die zunächst eine Ansammlung von Objekten zur Verfügung stellen und diese dann allmählich verbrauchen (ein weiteres Beispiel: Verteilen von Karten), gibt es mindestens zwei Möglichkeiten, sie in einem Feld zu speichern:

1. Man definiert ein Feld, das in seiner Struktur und Größe den Objekten entspricht und markiert verbrauchte Elemente mit einer bestimmten Zahl.

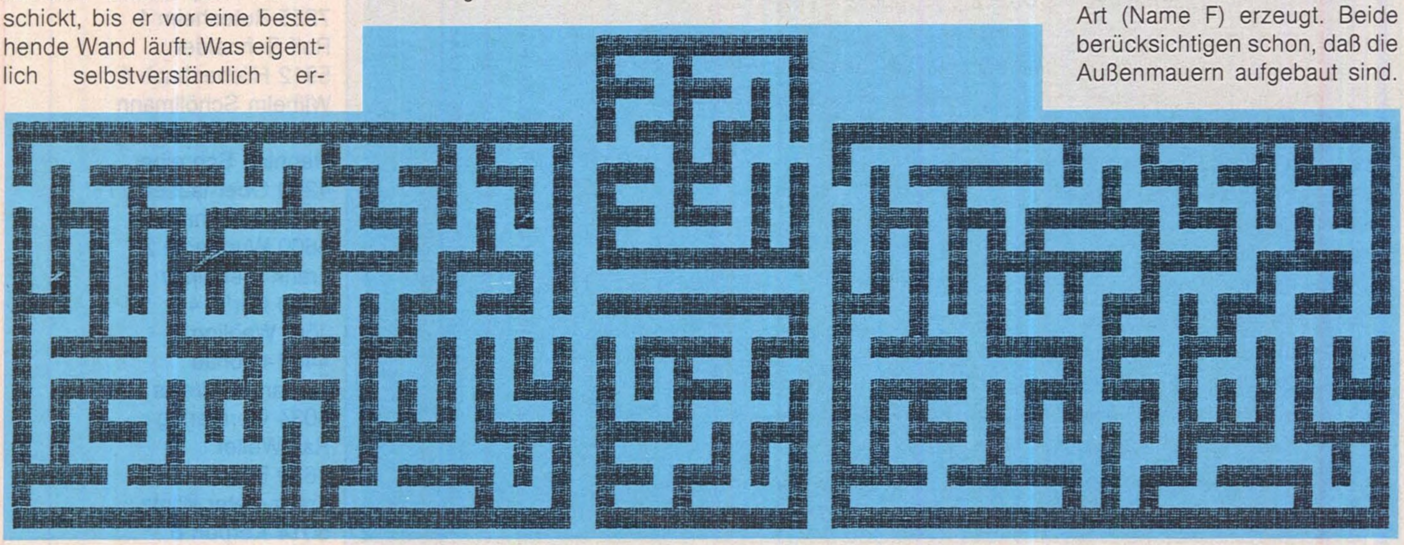
2. Man ordnet die noch freien Elemente hintereinander in einer Schlange an und numeriert sie jedesmal neu durch, nachdem ein Element entfernt wurde.

Das in 1. gemeinte Feld ist immer dann sehr nützlich, wenn das Programm wissen will, ob ein Objekt verbraucht ist oder nicht. Es schafft aber Probleme, wenn ein noch vorhandenes Objekt gesucht wird, nur noch wenige da sind und das Programm ständig Objekte vergeblich nach ihrem Vorhandensein befragt. Es kann dann sehr viel Zeit vergehen, bis ein Objekt gefunden wird. Aus dem Feld der 2. Art läßt sich jedoch schnell ein noch vorhandenes Objekt ermitteln.

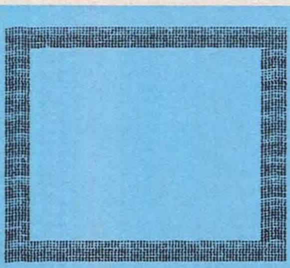
Vorbereitungen

Bei der Labyrinth-Erzeugung ist nur die Alternative Stelle besetzt – Stelle nicht besetzt wichtig. Die Elemente des zugehörigen Feldes der ersten Art erhalten dementsprechend die Werte Null oder Eins.

Nachdem in den Programmzeilen 10 und 20 die Länge und Breite des Labyrinths abgefragt werden, wird bis zur Programmzeile 200 sowohl ein Feld der ersten Art (Name G) als auch ein Feld der zweiten Art (Name F) erzeugt. Beide berücksichtigen schon, daß die Außenmauern aufgebaut sind.



```
5 LET D$="▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
10 INPUT X
20 INPUT Y
30 DIM F((X-1)*(Y-1),2)
40 LET FP=0
50 FOR I=2 TO X
60 FOR J=2 TO Y
70 LET FP=FP+1
80 LET F(FP,1)=I
90 LET F(FP,2)=J
100 NEXT J
110 NEXT I
120 DIM G(X+1,Y+1)
130 FOR I=1 TO X+1
140 LET G(I,1)=1
150 LET G(I,Y+1)=1
160 NEXT I
170 FOR I=1 TO Y+1
180 LET G(1,I)=1
190 LET G(X+1,I)=1
200 NEXT I
210 DIM R(4,2)
220 LET R(1,1)=-1
230 LET R(2,1)=1
240 LET R(3,2)=-1
250 LET R(4,2)=1
260 DIM W((X-1)*(Y-1)+1,2)
270 DIM X(X+1,Y+1)
300 FOR I=2 TO 2*(X+1)
310 PRINT LEFT$(D$,3);TAB(I);"●"
320 PRINT LEFT$(D$,2*(Y+1)+1);TAB(I);"●"
330 NEXT I
340 FOR I=2 TO 2*(Y+1)
350 PRINT LEFT$(D$,I+1);TAB(2);"●"
360 PRINT LEFT$(D$,I+1);TAB(2*(X+1));"●"
370 NEXT I
400 REM *** NEUE WAND ***
405 LET WP=1
410 FOR I=1 TO X+1
420 FOR J=1 TO Y+1
430 LET X(I,J)=0
440 NEXT J
450 NEXT I
460 LET ZS=INT(RND(1)*FP)+1
470 LET W(1,1)=F(ZS,1)
480 LET W(1,2)=F(ZS,2)
490 LET X(W(1,1),W(1,2))=1
500 LET WP=WP+1
510 LET ZR=INT(RND(1)*4)+1
520 LET W(WP,1)=W(WP-1,1)+R(ZR,1)
530 LET W(WP,2)=W(WP-1,2)+R(ZR,2)
540 LET XX=W(WP,1)
550 LET YY=W(WP,2)
600 REM *** FALSCHE RICHTUNG ODER ISOLATION ***
605 IF X(XX,YY)=1 THEN GOTO 510
680 PRINT LEFT$(D$,2*W(WP-1,2)+1);TAB(2*W(WP-1,1));"●"
690 PRINT LEFT$(D$,W(WP-1,2)+1+YY);TAB(W(WP-1,1)+XX);"●"
700 LET X(XX,YY)=1
710 IF G(XX,YY)=1 THEN GOTO 800
720 IF X(XX+1,YY)=0 OR X(XX-1,YY)=0 OR X(XX,YY-1)=0
    OR X(XX,YY+1)=0 THEN GOTO500
730 FOR I=WP-1 TO 1 STEP -1
740 LET XX=W(I,1)
750 LET YY=W(I,2)
760 PRINT LEFT$(D$,YY+W(I+1,2)+1);TAB(XX+W(I+1,1));" "
770 PRINT LEFT$(D$,2*YY+1);TAB(2*XX);" "
780 NEXT I
790 GOTO 400
800 REM *** WAND FERTIG ***
805 FOR I=1 TO WP-1
810 LET XX=W(I,1)
820 LET YY=W(I,2)
830 LET G(XX,YY)=1
840 NEXT I
850 FOR I=1 TO WP-1
860 FOR J=1 TO FP
870 IF F(J,1)=W(I,1) AND F(J,2)=W(I,2) THEN GOTO 900
880 NEXT J
890 GOTO 940
900 FOR K=J TO FP-1
910 LET F(K,1)=F(K+1,1)
920 LET F(K,2)=F(K+1,2)
930 NEXT K
940 NEXT I
950 LET FP=FP-WP+1
960 IF FP>0 THEN GOTO 400
1000 REM *** EINGANG, AUSGANG ***
1005 LET YY=2*(INT(RND(1)*(Y-1))+1)+1
1010 LET XX=2
1020 PRINT LEFT$(D$,YY+1);TAB(XX);" "
1030 LET YY=2*(INT(RND(1)*(Y-1))+1)+1
1040 LET XX=2*(X+1)
1050 PRINT LEFT$(D$,YY+1);TAB(XX);" "
```



1	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	1	1	1	1	1

Feld G

2	2
2	3
2	4
2	5
3	2
3	3
3	4
3	5
4	2
4	3
4	4
4	5
5	2
5	3
5	4
5	5

Feld F

Die Elemente von F geben jeweils zuerst die Spaltenzahl und dann die Zeilenzahl der noch freien Stellen an.

Das Labyrinth nach dem Aufbau der Außenmauern

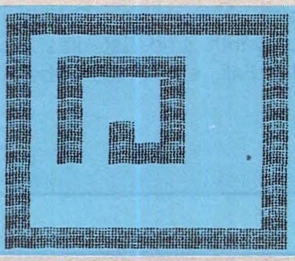
Nebenbei wurde der Variablen FP, die die Anzahl der freien Stellen angibt, in Zeile 70 allmählich der richtige Wert beigebracht. In den Programmzeilen 210 bis 250 wird das Feld R konstruiert, das die vier Himmelsrichtungen wiedergibt. In den Programmzeilen 260 und 270 werden die Felder W und X vereinbart. Sie entsprechen den Feldern F und G, beziehen sich aber nur auf die gerade entstandene Wand. Es folgt bis Programmzeile 370 die Zeichnung der Außenmauern.

Ab Programmzeile 405 beginnen die Aufbauarbeiten der Innenwände. Zunächst wird die Anzahl der Wandpunkte auf Eins gesetzt und das Feld X gelöscht. Programmzeile 460 bestimmt zufällig, an welcher von den noch freien Stellen des Labyrinths die neue Mauer begonnen werden soll. Bis Programmzeile 490 wird diese Wahl in den Feldern W und X berücksichtigt.

Der nächste Wandpunkt wird vom Programm vorgeschlagen, indem eine von vier Richtungen zufällig ausgewählt wird. Dies wird in Programmzeile 605 abgelehnt, wenn die Wand damit auf sich selbst treffen würde; in diesem Fall wird die Richtungswahl wiederholt. Ist dies nicht nötig, wird in den Programmzeilen 680 700 das neue Stück gezeichnet und im Feld X vermerkt. Programmzeile 710 beendet die Aufbauarbeiten, falls eine bereits bestehende andere Wand angetroffen wird.

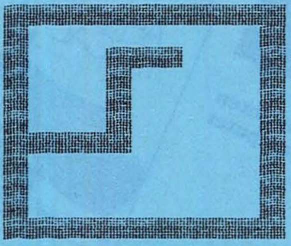
Mühsame Aufbauarbeiten

Ein Sonderfall wird in Programmzeile 720 behandelt. Es kann nämlich vorkommen, daß eine Wand sich im nächsten



Sackgasse

Schritt unvermeidlich selbst in den Schwanz beißt. Um aus dieser Sackgasse wieder herauszukommen, genügt es nicht, den letzten Schritt in die falsche Richtung zurückzuneh-



1	1	1	1	1	1
1	0	1	1	0	1
1	0	1	0	0	1
1	1	1	0	0	1
1	0	0	0	0	1
1	1	1	1	1	1

Feld G

2	2
2	3
2	5
3	5
4	3
4	4
4	5
5	2
5	3
5	4
5	5

Feld F

Scan: S. Höltgen


```
   5 REM
 310 PRINT AT 2,i;"■"
 320 PRINT AT 2*(y+1),i;"■"
 350 PRINT AT i,2;"■"
 360 PRINT AT i,2*(x+1);"■"
 460 LET zs=INT (RND*fp)+1
 510 LET zr=INT (RND*4)+1
 680 PRINT AT 2*w(wp-1,2),2*w(wp
-1,1);"■"
 690 PRINT AT w(wp-1,2)+yy,w(wp-
1,1)+xx;"■"
 760 PRINT AT yy+w(i+1,2),xx+w(i
+1,1);" "
 770 PRINT AT 2*yy,2*xx;" "
1005 LET yy=2*(INT (RND*(y-1))+1
)+1
1020 PRINT AT yy,xx;" "
1030 LET yy=2*(INT (RND*(y-1))+1
)+1
1050 PRINT AT yy,xx;" "
```

Änderungen für ZX 81 und Spectrum

```
5 REM
121 FOR I=2 TO X
122 FOR J=2 TO Y
123 G(I,J)=0
124 NEXT J
125 NEXT I
211 FOR I=1 TO 4
212 R(I,1)=0
213 R(I,2)=0
214 NEXT I
310 POSITION I,2:PRINT CHR$(160)
320 POSITION I,2*(Y+1):PRINT CHR$(160)
350 POSITION 2,I:PRINT CHR$(160)
360 POSITION 2*(X+1),I:PRINT CHR$(160)
680 POSITION 2*W(WP-1,1),2*W(WP-1,2):PRINT CHR$(160)
690 POSITION W(WP-1,1)+XX,W(WP-1,2)+YY:PRINT CHR$(160)
760 POSITION XX+W(I+1,1),YY+W(I+1,2):PRINT " "
770 POSITION 2*XX,2*YY:PRINT " "
1020 POSITION XX,YY:PRINT " "
1050 POSITION XX,YY:PRINT " "
2000 POSITION 0,23
```

Änderungen für Atari

men. Der Einfachheit halber wird dann gleich die ganze Wand eingerissen.

Die nötigen Korrekturen an F, G und FP nach einem erfolgreichen Wandabschluß werden in den Programmzeilen 805 bis 950 vorgenommen.

Die erste Innenwand

Programmzeile 960 stellt fest, ob noch freie Stellen im Labyrinth sind. Ist dies nicht der Fall, erhält das Labyrinth links und rechts einen Eingang und einen Ausgang.

Hinweise: Das Programm läßt sich noch erheblich verkürzen und kommt mit weniger Feldern aus, wenn man direkt den Bildspeicher abfragt, um über freie oder besetzte Stellen Auskunft zu bekommen. Dies kann entweder mit Hilfe der PEEK-Funktion erfolgen oder mit SCREEN$ beim Spectrum, GCHAR beim TI-99/4 und LOCATE beim Atari).

Horst Brand

```
5 CALL CHAR(33,"FFFFFFFFFFFF
FFFF")
10 INPUT IX
25 CALL CLEAR
30 DIM F(130,2)
50 FOR I=2 TO IX
120 DIM G(15,12)
121 FOR I=2 TO IX
122 FOR J=2 TO Y
123 LET G(I,J)=0
124 NEXT J
125 NEXT I
130 FOR I=1 TO IX+1
190 LET G(IX+1,I)=1
211 FOR I=1 TO 4
212 R(I,1)=0
213 R(I,2)=0
214 NEXT I
260 DIM W(130,2)
270 DIM X(15,12)
300 FOR I=2 TO 2*(IX+1)
310 CALL HCHAR(2,I,33,1)
320 CALL HCHAR(2*(Y+1),I,33,
1)
350 CALL HCHAR(I,2,33,1)
360 CALL HCHAR(I,2*(IX+1),33
,1)
410 FOR I=1 TO IX+1
460 LET ZS=INT(RND*FP)+1
510 LET ZR=INT(RND*4)+1
605 IF X(XX,YY)=1 THEN 510
680 CALL HCHAR(2*W(WP-1,2),2
*W(WP-1,1),33,1)
690 CALL HCHAR(W(WP-1,2)+YY,
W(WP-1,1)+XX,33,1)
710 IF G(XX,YY)=1 THEN 800
720 IF (X(XX+1,YY)+X(XX-1,YY
)+X(XX,YY-1)+X(XX,YY+1))<4 T
HEN 500
760 CALL HCHAR(YY+W(I+1,2),X
X+W(I+1,1),32,1)
770 CALL HCHAR(2*YY,2*XX,32,
1)
870 IF (ABS(F(J,1)-W(I,1))+A
BS(F(J,2)-W(I,2)))=0 THEN 90
0
960 IF FP>0 THEN 400
1005 LET YY=2*(INT(RND*(Y-1)
)+1)+1
1020 CALL HCHAR(YY,XX,32,1)
1030 LET YY=2*(INT(RND*(Y-1)
)+1)+1
1040 LET XX=2*(IX+1)
1050 CALL HCHAR(YY,XX,32,1)
1060 GOTO 1060
```

Änderungen für TI-99/4A

Scan: S. Höltgen

Feuer

Ein auch unter dem Namen „Schwimmen“ bekanntes Kartenspiel für drei Spieler, von denen zwei durch den TI-99/4A vertreten werden

Jeder Spieler bekommt drei Karten und versucht, möglichst viele Punkte von einer Farbe zu sammeln. Um dieses Ziel zu erreichen, dürfen Karten „aus der Mitte“ genommen und eigene abgelegt werden. Wer von drei Pfennigen, dem Kapital jedes Spielers, alle verloren hat, der „schwimmt“ und darf nur noch einmal mitspielen; bei erneutem Verlust scheidet er aus. Viel Spaß, und kommen Sie nicht zu oft ins Schwimmen!

Michael Silberberg

```
100 REM ********************

110 REM *    SCHWIMMEN     *

120 REM *  COPYRIGHT BY    *

130 REM *     MICHAEL      *

140 REM *   SILBERBERG     *

150 REM * BERLINER STR.10 *

160 REM *     5657 HAAN    *

170 REM * 5. FEBRUAR 1984 *

180 REM ********************

190 REM  ALLE REM's KOENNEN
WEGGELASSEN WERDEN
200 FOR N=1 TO 12
210 CALL COLOR(N,16,1)
220 NEXT N
230 CALL SCREEN(2)
240 REM  PROGRAMMSTART
250 CALL CLEAR
260 CALL CHAR(124,"101010109
254381")
270 CALL CHAR(94,"081C2A4908
080808")
280 CALL CHAR(95,"000000FFFF
")
290 PRINT TAB(10);"SCHWIMMEN
"::::"WUENSCHEN SIE DIE SPIE
L-     REGELN ZU LESEN ?"
::::::
300 CALL KEY(0,MS,ZX)
310 IF ZX=0 THEN 300
320 IF MS=78 THEN 540
330 IF MS<>74 THEN 300
340 CALL CLEAR
350 PRINT TAB(10);"SCHWIMMEN
":::"EIN KARTENSPIEL FUER DR
EI    PERSONEN,WOVON DER
TI ZWEI  UEBERNIMMT"
360 PRINT "ZIEL DES SPIELES
IST ES,    MOEGLICHST VIELE
PUNKTE VON EINER KARTENF
ARBE ZU BE-"
370 PRINT "KOMMEN.SIE DUERFE
N EINE      KARTE AUS DER MIT
TE NEHMEN, MUESSEN DAFUE
R ABER EINE"
380 PRINT "ANDERE KARTE IN D
IE MITTE   LEGEN.WENN SIE KE
INE KARTEN MEHR TAUSCHEN
 WOLLEN,DANN"
390 PRINT "MUESSEN SIE >ENTE
R< DRUECKEN.................
............"::::
400 PRINT TAB(15);"TASTE DRU
ECKEN"
410 CALL KEY(0,MS,ZX)
420 IF ZX=0 THEN 410
430 CALL CLEAR
440 PRINT "WENN SIE >ENTER<
GEDRUECKT  HABEN,WIRD NOCH E
INE RUNDE  GESPIELT."
450 PRINT "DIES GILT AUCH,WE
NN DER TI  DIE LETZTE RUNDE
ANKUENDIGT."
460 PRINT "WER DIE WENIGSTEN
 PUNKTE    HAT,BEKOMMT EINEN
 SEINER    DREI PFENNIGE
 ABGEZOGEN."
470 PRINT "WER KEINE PFENNIG
E MEHR HAT,DARF 'schwimmen'.
D.H.,ER    SPIELT NOCH S
OLANGE MIT,BIS"
480 PRINT "ER NOCHMALS VERLO
REN HAT.   DERJENIGE,DER VIE
RMAL VER-  LOREN HAT,SCH
EIDET AUS."
490 PRINT "WENN AUF DEM KART
ENHAUFEN   DREI KARTEN MIT D
ER GLEICHENFARBE LIEGEN,
KOENNEN SIE PER"
500 PRINT ">A< ALLE KARTEN A
USWECHSELN!";TAB(8);"*VIEL G
LUECK*";"p.s.SIE SIND DE
R SPIELER          NR.3"::
510 PRINT TAB(15);"TASTE DRU
ECKEN"
520 CALL KEY(0,MS,ZX)
530 IF ZX=0 THEN 520
540 CALL CLEAR
550 FOR N=1 TO 3
560 SCH(N)=3
570 NEXT N
580 RANDOMIZE
590 M$="ICH MISCHE !"
600 CALL CLEAR
610 MI=MI+1
620 IF MI<4 THEN 640
630 MI=1
640 Y=10
650 X=10
660 GOSUB 1830
670 REM  EINLESEN DER DATAS
680 DATA kreuz,pik,herz,karo
690 RESTORE 680
700 FOR N=1 TO 4
710 READ F$(N)
720 NEXT N
730 DATA 7,8,9,10,bube,dame,
koenig,ass
740 FOR N=1 TO 8
750 READ W$(N)
760 T$(N)=W$(N)
770 NEXT N
780 DATA 7,8,9,10,10,10,10,1
1
790 FOR N=1 TO 8
800 READ WW(N)
810 NEXT N
820 FOR N=1 TO 4
830 FOR M=1 TO 8
840 WE$(N,M)=W$(M)
850 NEXT M
860 NEXT N
870 REM  'MISCHEN'
880 FOR N=4 TO 1 STEP -1
890 FOR M=1 TO 3
900 NZ=INT(RND*4+1)
910 MZ=INT(RND*8+1)
920 IF WE$(NZ,MZ)="" THEN 90
0
930 SP$(N,M)=WE$(NZ,MZ)
940 PS$(N,M)=F$(NZ)
950 WE$(NZ,MZ)=""
960 NEXT M
970 NEXT N
980 CALL CLEAR
990 REM  AUSDRUCK DES KARTEN
STAPELS
1000 REM  ACHTUNG ! DIE ANZA
HL DER DOPPELPUNKTE genau BE
ACHTEN
1010 FOR N=1 TO 3
1020 PRINT TAB(10);N;PS$(4,N
```

Scan: S. Höltgen


```
);"-";SP$(4,N)
1030 NEXT N
1040 PRINT ::::::
1050 FOR N=1 TO 3
1060 PRINT TAB(10);N;PS$(3,N
);"-";SP$(3,N)
1070 NEXT N
1080 PRINT ::
1090 CALL HCHAR(15,1,95,32)
1100 M$="IHRE KARTEN :"
1110 Y=16
1120 X=7
1130 GOSUB 1830
1140 M$="KARTENHAUFEN ^"
1150 Y=14
1160 X=6
1170 GOSUB 1830
1180 MA=MI
1190 REM  WER FAENGT AN ?
1200 ON MA GOSUB 4030,4070,4
110
1210 ON MI GOTO 1690,1690,12
20
1220 DATA 3,2,WELCHE KARTE W
OLLEN SIE,5,2,NEHMEN ? (1-3)
1230 RESTORE 1220
1240 MA=1
1250 IF LE(3)=1 THEN 2730
1260 IF SCH(3)=-1 THEN 1690
1270 M$=""
1280 GOSUB 1830
1290 FOR N=1 TO 2
1300 READ Y,X,M$
1310 GOSUB 1810
1320 NEXT N
1330 REM  ABFRAGE IHRER EING
ABE
1340 CALL KEY(0,MS,ZX)
1350 IF ZX=0 THEN 1340
1360 IF MS=65 THEN 4160
1370 IF MS<>13 THEN 1400
1380 LE(3)=1
1390 GOTO 1690
1400 MS=MS-48
1410 IF (MS<1)+(MS>3)THEN 13
40
1420 ZM=MS
1430 K$=PS$(4,MS)
1440 K1$=SP$(4,MS)
1450 CALL HCHAR(9+MS,1,32,32
)
1460 Y=5
1470 X=2
1480 M$="WEGLEGEN ?(1-3)"
1490 GOSUB 1810
1500 CALL KEY(0,MS,ZX)
1510 IF ZX=0 THEN 1500
1520 MS=MS-48
1530 IF (MS<1)+(MS>3)THEN 15
00
1540 CALL HCHAR(1,1,32,160)
1550 PS$(4,ZM)=PS$(3,MS)
1560 SP$(4,ZM)=SP$(3,MS)
1570 Y=9+ZM
1580 X=12
1590 M$=STR$(ZM)&" "&PS$(3,M
S)&"-"&SP$(3,MS)
1600 GOSUB 1830
1610 PS$(3,MS)=K$
1620 SP$(3,MS)=K1$
1630 Y=18+MS
1640 X=12
1650 CALL HCHAR(Y,1,32,32)
1660 M$=STR$(MS)&" "&K$&"-"&
K1$
1670 GOSUB 1810
1680 REM  COMPUTERSPIEL
1690 FOR N=MA TO MA-(MA=1)
1700 IF SCH(N)=-1 THEN 1790
1710 IF LE(N)=1 THEN 2730
1720 FOR M=1 TO 3
1730 P$(M)=PS$(N,M)
1740 NEXT M
1750 IF (P$(1)=P$(2))*(P$(2)
=P$(3))THEN 1890
1760 GOTO 4380
1770 WEG=0
1780 GOTO 2500
1790 NEXT N
1800 GOTO 1220
1810 GOTO 1840
1820 REM  ERSATZ FUER DISPLA
Y AT
1830 CALL HCHAR(1,1,32,160)
1840 FOR AS=1 TO LEN(M$)
1850 CALL HCHAR(Y,X+AS,ASC(S
EG$(M$,AS,1)))
1860 NEXT AS
1870 RETURN
1880 REM   UEBERPRUEFEN,OB N
OCH EINE KARTE AUF DEM HAUFE
N LIEGT,DIE MAN TAUSCHEN
 KOENNTE,OBWOHL 4 GLEICHE KA
RTEN VORHANDEN SIND
1890 M$=P$(1)
1900 FOR M=1 TO 3
1910 IF M$=PS$(4,M)THEN 2020
1920 NEXT M
1930 Y=3
1940 X=2
1950 M$="LETZTE RUNDE !"
1960 GOSUB 1830
1970 LE(N)=1
1980 FOR M=0 TO 30
1990 CALL SOUND(-100,-1,M)
2000 NEXT M
2010 GOTO 1790
2020 FOR M=1 TO 3
2030 P$(M)=SP$(N,M)
2040 NEXT M
2050 FOR M=1 TO 3
2060 FOR AS=1 TO 8
2070 IF P$(M)<>T$(AS)THEN 20
90
2080 QW(M)=WW(AS)
2090 NEXT AS
2100 NEXT M
2110 FOR M=1 TO 3
2120 FOR AS=1 TO 8
2130 IF SP$(4,M)<>T$(AS)THEN
 2150
2140 WQ(M)=WW(AS)
2150 NEXT AS
2160 NEXT M
2170 FOR M=1 TO 3
2180 FOR AS=1 TO 3
2190 IF QW(M)<WW(AS)THEN 224
0
2200 NEXT AS
2210 NEXT M
2220 GOTO 1930
2230 GOTO 1790
2240 NEH=AS
2250 WEG=M
2260 Y=2
2270 X=2
2280 M$="DER SPIELER TI"&STR
$(N)&" NIMMT"
2290 GOSUB 1830
2300 Y=4
2310 X=2
2320 M$="DIE KARTE NR."&STR$
(NEH)
2330 GOSUB 1810
2340 FOR AS=1 TO 200
2350 NEXT AS
2360 REM  AUSWECHSELN DER KA
RTEN
2370 K$=PS$(4,NEH)
2380 K1$=SP$(4,NEH)
2390 CALL HCHAR(9+NEH,1,32,3
2)
2400 PS$(4,NEH)=PS$(N,WEG)
2410 SP$(4,NEH)=SP$(N,WEG)
2420 M$=STR$(NEH)&" "&PS$(4,
NEH)&"-"&SP$(4,NEH)
2430 Y=9+NEH
2440 X=12
2450 GOSUB 1810
2460 PS$(N,WEG)=K$
2470 SP$(N,WEG)=K1$
2480 GOTO 1790
2490 REM  WELCHE KARTE SOLL
WEGGELEGT WERDEN ?
2500 IF (P$(1)<>P$(2))*(P$(1
)=P$(3))THEN 2510 ELSE 2520
2510 WEG=2
2520 IF (P$(1)<>P$(3))*(P$(1
)=P$(2))THEN 2530 ELSE 2540
2530 WEG=3
2540 IF (P$(1)<>P$(2))*(P$(2
)=P$(3))THEN 2550 ELSE 2560
2550 WEG=1
2560 IF WEG=0 THEN 2700
2570 IF WEG=2 THEN 2590
2580 K$=PS$(N,2)
2590 IF WEG=1 THEN 2610
2600 K$=PS$(N,1)
2610 IF WEG=3 THEN 2630
2620 K$=PS$(N,3)
2630 FOR M=1 TO 3
2640 IF K$=PS$(4,M)THEN 2650
 ELSE 2670
2650 NEH=M
2660 GOTO 2260
2670 NEXT M
2680 NEH=INT(RND*3+1)
2690 GOTO 2260
2700 WEG=INT(RND*3+1)
2710 NEH=INT(RND*3+1)
2720 GOTO 2260
```

Scan: S. Höltgen


```
2730 CALL CLEAR
2740 REM  SPIELENDE
2750 PRINT TAB(10);"AUSWERTU
NG:"::
2760 FOR N=1 TO 3
2770 IF SCH(N)=-1 THEN 2830
2780 PRINT "SPIELER NR.";N;"
 HAT FOLGENDE KARTEN:"
2790 FOR M=1 TO 3
2800 PRINT PS$(N,M);"-";SP$(
N,M)
2810 NEXT M
2820 PRINT :
2830 NEXT N
2840 FOR N=1 TO 111
2850 CALL SOUND(-N,N+110,0)
2860 NEXT N
2870 FOR N=1 TO 3
2880 IF SCH(N)=-1 THEN 3330
2890 IF (PS$(N,1)=PS$(N,2))*
(PS$(N,2)=PS$(N,3))THEN 2900
 ELSE 2970
2900 FOR M=1 TO 3
2910 FOR AS=1 TO 8
2920 IF SP$(N,M)<>T$(AS)THEN
 2940
2930 PU(N)=PU(N)+WW(AS)
2940 NEXT AS
2950 NEXT M
2960 GOTO 3330
2970 IF PS$(N,1)=PS$(N,2)THE
N 2980 ELSE 3050
2980 FOR M=1 TO 2
2990 FOR AS=1 TO 8
3000 IF SP$(N,M)<>T$(AS)THEN
 3020
3010 PU(N)=PU(N)+WW(AS)
3020 NEXT AS
3030 NEXT M
3040 GOTO 3330
3050 IF PS$(N,1)=PS$(N,3)THE
N 3060 ELSE 3130
3060 FOR M=1 TO 3 STEP 2
3070 FOR AS=1 TO 8
3080 IF SP$(N,M)<>T$(AS)THEN
 3100
3090 PU(N)=PU(N)+WW(AS)
3100 NEXT AS
3110 NEXT M
3120 GOTO 3330
3130 IF PS$(N,2)=PS$(N,3)THE
N 3140 ELSE 3210
3140 FOR M=2 TO 3
3150 FOR AS=1 TO 8
3160 IF SP$(N,M)<>T$(AS)THEN
 3180
3170 PU(N)=PU(N)+WW(AS)
3180 NEXT AS
3190 NEXT M
3200 GOTO 3330
3210 FOR M=1 TO 3
3220 FOR AS=1 TO 8
3230 IF SP$(N,M)<>T$(AS)THEN
 3250
3240 RT(N,M)=WW(AS)
3250 NEXT AS
3260 NEXT M
3270 IF (RT(N,1)>RT(N,2))*(R
T(N,1)>RT(N,3))THEN 3280 ELS
E 3290
3280 PU(N)=RT(N,1)
3290 IF (RT(N,2)>RT(N,1))*(R
T(N,2)>RT(N,3))THEN 3300 ELS
E 3310
3300 PU(N)=RT(N,2)
3310 IF PU(N)<>0 THEN 3330
3320 PU(N)=RT(N,3)
3330 NEXT N
3340 CALL CLEAR
3350 PRINT TAB(10);"PUNKTE:"
::
3360 FOR N=1 TO 3
3370 IF SCH(N)=-1 THEN 3390
3380 PRINT "SPIELER NR.";N;"
 HAT ";PU(N);" PUNKTE.":::
3390 NEXT N
3400 IF (SCH(1)=-1)+(SCH(2)=
-1)THEN 3790
3410 IF (PU(1)<=PU(2))*(PU(1
)<=PU(3))THEN 3420 ELSE 3430
3420 SCH(1)=SCH(1)-1
3430 IF (PU(2)<=PU(3))*(PU(2
)<=PU(1))THEN 3440 ELSE 3450
3440 SCH(2)=SCH(2)-1
3450 IF (PU(3)<=PU(2))*(PU(3
)<=PU(1))THEN 3460 ELSE 3470
3460 SCH(3)=SCH(3)-1
3470 FOR N=1 TO 3
3480 PU(N)=0
3490 LE(N)=0
3500 FOR M=1 TO 3
3510 RT(N,M)=0
3520 NEXT M
3530 NEXT N
3540 FOR N=1 TO 111
3550 CALL SOUND(-N,RND*N+110
,5)
3560 NEXT N
3570 PRINT TAB(5);"PFENNIGVE
RTEILUNG"::
3580 FOR N=1 TO 3
3590 PRINT "SPIELER NR.";N;"
 HAT "
3600 IF SCH(N)<>-1 THEN 3620
3610 PRINT "KEINE PFENNIGE M
EHR UND IST AUSGESCHIEDEN.":
3620 IF SCH(N)<>0 THEN 3640
3630 PRINT "NULL PFENNIGE UN
D IST AM     SCHWIMMEN.":
3640 IF SCH(N)<2 THEN 3660
3650 PRINT "NOCH";SCH(N);"PF
ENNIGE."
3660 IF SCH(N)<>1 THEN 3680
3670 PRINT "NOCH EINEN PFENN
IG."
3680 NEXT N
3690 FOR N=1 TO 111
3700 CALL SOUND(-100,-1,N/4)
3710 NEXT N
3720 IF SCH(3)=-1 THEN 3750
3730 IF (SCH(1)=-1)*(SCH(2)=
-1)THEN 3920
3740 GOTO 580
3750 CALL CLEAR
3760 PRINT "DA SIE KEIN GELD
 MEHR HABEN,MACHT ES UNS KEI
NEN SPASS,  WEITER ZU SP
IELEN."
3770 PRINT :::"AUF WIEDERSEH
N.":::
3780 END
3790 IF SCH(1)=-1 THEN 3860
3800 IF PU(1)=PU(3)THEN 3970
3810 IF PU(1)<PU(3)THEN 3840
3820 SCH(3)=SCH(3)-1
3830 GOTO 3470
3840 SCH(1)=SCH(1)-1
3850 GOTO 3470
3860 IF PU(2)=PU(3)THEN 4000
3870 IF PU(2)<PU(3)THEN 3900
3880 SCH(3)=SCH(3)-1
3890 GOTO 3470
3900 SCH(2)=SCH(2)-1
3910 GOTO 3470
3920 CALL CLEAR
3930 PRINT "NA GUT,SIE HABEN
 GEWONNEN."::"JETZT MOEGEN W
IR NICHT MEHR."
3940 PRINT "HOFFENTLICH SEHE
N WIR UNS    ERST WIEDER,WENN
 SIE DAS "
3950 PRINT "FALSCHSPIELEN VE
RLERNT       HABEN."
3960 END
3970 SCH(1)=SCH(1)-1
3980 SCH(3)=SCH(3)-1
3990 GOTO 3470
4000 SCH(2)=SCH(2)-1
4010 SCH(3)=SCH(3)-1
4020 GOTO 3470
4030 IF SCH(1)<>-1 THEN 4060
4040 MA=2
4050 MI=MA
4060 RETURN
4070 IF SCH(2)<>-1 THEN 4100
4080 MA=3
4090 MI=MA
4100 RETURN
4110 IF SCH(3)<>-1 THEN 4140
4120 MA=1
4130 MI=MA
4140 RETURN
4150 REM   KOENNEN ALLE GETA
USCHT WERDEN ? (SPIELER)
4160 IF (PS$(4,1)=PS$(4,2))*
(PS$(4,2)=PS$(4,3))THEN 4170
 ELSE 1340
4170 FOR N=1 TO 3
4180 K$=SP$(4,N)
4190 K1$=PS$(4,N)
4200 SP$(4,N)=SP$(3,N)
4210 PS$(4,N)=PS$(3,N)
4220 SP$(3,N)=K$
4230 PS$(3,N)=K1$
4240 NEXT N
4250 FOR N=1 TO 3
4260 Y=9+N
4270 CALL HCHAR(Y,1,32,32)
4280 X=12
4290 M$=STR$(N)&" "&PS$(4,N)
&"-"&SP$(4,N)
```

Scan: S. Höltgen


```
4300 GOSUB 1830
4310 Y=18+N
4320 CALL HCHAR(Y,1,32,32)
4330 M$=STR$(N)&" "&PS$(3,N)
&"-"&SP$(3,N)
4340 GOSUB 1830
4350 NEXT N
4360 GOTO 1690
4370 REM   KOENNEN ALLE GETA
USCHT WERDEN ? (COMPUTER)
4380 IF (PS$(4,1)=PS$(4,2))*
(PS$(4,2)=PS$(4,3))THEN 4390
 ELSE 1770
4390 FOR MR=1 TO 3
4400 K$=SP$(4,MR)
4410 K1$=PS$(4,MR)
4420 SP$(4,MR)=SP$(N,MR)
4430 PS$(4,MR)=PS$(N,MR)
4440 SP$(N,MR)=K$
4450 PS$(N,MR)=K1$
4460 NEXT MR
4470 M$="SPIELER TI"&STR$(N)
&" TAUSCHT ALLE !"
4480 Y=3
4490 X=1
4500 GOSUB 1830
4510 FOR MR=1 TO 3
4520 Y=9+MR
4530 CALL HCHAR(Y,1,32,32)
4540 X=12
4550 M$=STR$(MR)&" "&PS$(4,M
R)&"-"&SP$(4,MR)
4560 GOSUB 1840
4570 NEXT MR
4580 GOTO 1790
```

Entenjagd

Der Colour-Genie läßt Enten über den Bildschirm flattern. Sie sind ein Jäger, der mit einem Gewehr und seinem treuen Hund zur Jagd gegangen ist

Das Programm wird mit RUN gestartet. Zunächst werden die freidefinierbaren Zeichen geladen. Danach kommt die Startgrafik und die Frage, ob man die Anleitung wünscht oder sofort den Enten an den Kragen gehen will.

Das eigentliche Programm beginnt mit dem Einmarsch des Jägers, dem sein Hund auf den Fersen folgt.

Nachdem er sich am linken Bildrand postiert hat, wird es ernst: Sofort erscheint die erste Ente am Himmel. Flughöhe und Flugrichtung sind durch Zufallsgenerator bestimmt. Sie haben nun die Möglichkeit, das Gewehr mit der Pfeil-oben-Taste zu heben oder mit der Pfeil-unten-Taste zu senken. Der Schuß geht bei Druck auf die Space-Taste los.

Waidmannsheil

Trifft man eine Ente, segelt sie zu Boden, und Ihr treuer Hund apportiert sie, was zehn Punkte bringt. Bei einem Luftloch (und solche gibt's am Anfang bestimmt viele) weint der Himmel, und fünf Strafpunkte werden auch noch abgezogen.

Um das stark strapazierte Punkte- und Munitionskonto etwas aufzufrischen, taucht zufällig am rechten Bildrand ein laufender Hirsch auf. Wenn man schnell genug ist, kann man fünfzig Bonuspunkte und zwei Schuß gewinnen. Sind alle zwanzig Patronen verbraucht, ist das Spiel zu Ende.

Falls man gut war, darf man sich in die High-Score-Tabelle eintragen. Ein neues Spiel beginnt mit Druck auf eine beliebige Taste.

Zum Programm

Die Besitzer eines Colour-Genie mit 16-KByte-RAM müssen unbedingt bei dem Einschalten ihres Computers die MOD-SEL-Taste drücken, sonst gibt es einen Memory-Overflow-Error.

Die Grafikzeichen in den Programmzeilen 700–720, 7150–7180, 7720–7735 sind nach Drücken der MOD-SEL-Taste unter den entsprechenden Buchstaben erreichbar.

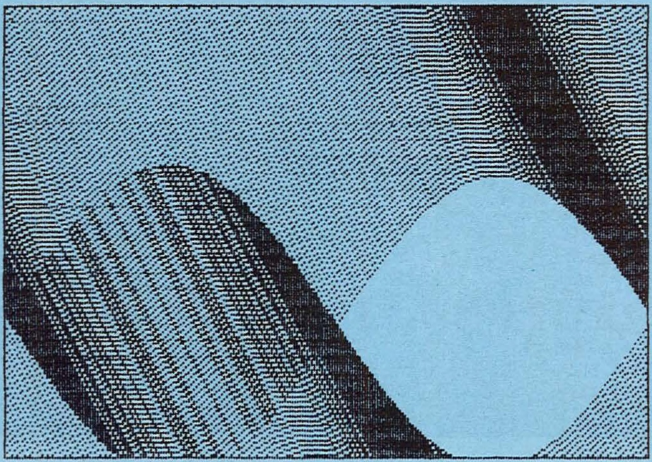
Sie brauchen nicht alle REMs mit einzugeben, nur solche, die als Überschrift für einen Block dienen, da das der Programmeinsprung ist.

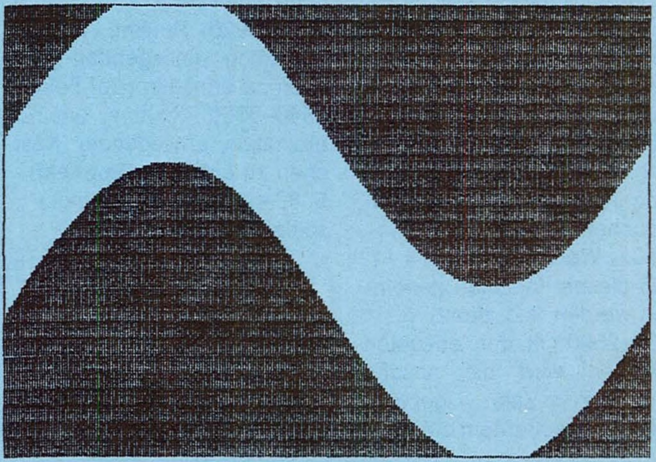
In Zeile 7300 muß bei der PRINT-USING-Anweisung das Pfundzeichen durch # ersetzt werden. Wem der Hirsch zu kurz erscheint, kann die Entfernung, die er zurücklegt, in Zeile 6080 vergrößern; sie ist in der Variablen HF gespeichert.

Dieter Bruckelt

```
10 '***********************************
20 '********** Duck shotting **********
30 '****** (c) Dieter Bruckelt ********
40 '** Bei der 16k Byte Version vor ***
50 '** einschalten die MOD SEL-Taste **
60 '********** druecken ! ************
70 '***********************************
90 DEFINTA-Z
100 CLEAR1000:CLS:PRINT@490,"(c) Dieter Bruckelt":GOSUB5000:GOSUB7500:GOSUB7700
105 IF I$="A"THEN GOSUB8000
499 '
500 ' INITIALLISIERUNG
501 '
510 Z1=0:EF=0:MU=20:RANDOM:CLS:COLOUR16:GP=0:PU=0
520 GOSUB6000
```

Scan: S. Höltgen





schon bei WEITER beschrieben. Um das Programm einzugeben, tippt man zuerst den Hexloader ein, dann startet man ihn, und es werden die sedezimalen Zahlen des Hexdumps und seine dezimalen Prüfsummen abgefragt. Hat man sie eingegeben, speichert man das Hardcopy-Programm mit CSAVE"HARDCOPY", 20000,&H4ECB,20000 ab. Danach kann man es testen, indem man den PMODE4-Bildschirm mit einer Grafik versieht, zum Beispiel durch das BASIC-Programm zum Beispielausdruck, und EXEC 20000 eingibt. Wenn der Drucker nur Buchstaben von sich geben sollte, gibt man PRINT#-2,CHR$(8):EXEC 20000 ein.

Wolfgang Sohrt

Paper-Boxing

Ein Denkspiel mit dem TI-99/4A in der Grundversion als Gegenspieler

Spielregeln

Jeder Spieler malt ein quadratisches Spielfeld mit 4 × 4 Kästchen. In die Kästchen verteilt man nach Belieben die Zahlen von 1 bis 15, wobei das Kästchen links oben frei bleiben muß, es wird das Startfeld. Wenn beide Spieler fertig sind, werden beide Spielfelder offengelegt. Die beiden Zahlen im Kästchen rechts unten werden addiert. Ist die Summe eine gerade Zahl, so beginnt der Spieler, ist sie ungerade, so beginnt der Computer.

Im Startfeld steht nun der Boxer, ein rot umrahmtes weißes Kästchen. Er muß nun in ein angrenzendes Kästchen ziehen, senkrecht, waagrecht oder diagonal. Dasselbe macht daraufhin der andere Spieler. Die beiden Zahlen in den erreichten Kästchen werden verglichen, und wer die höchste Zahl hat, bekommt einen Punkt und zieht beim nächsten Zug zuerst. Sind beide Zahlen gleich, so bekommt keiner einen Punkt, und derselbe Spieler zieht weiter als erster.

Schon betretene Kästchen werden mit einem roten Quadrat markiert und dürfen nicht mehr benutzt werden. Sind alle 15 Zahlen rot markiert, ist Spielende. Wer die meisten Punkte hat, ist Sieger. Bei Punktgleichheit ist unentschieden. Man kann aber auch durch k.o. verlieren, nämlich dann, wenn man sich vergaloppiert hat und nicht mehr ziehen kann.

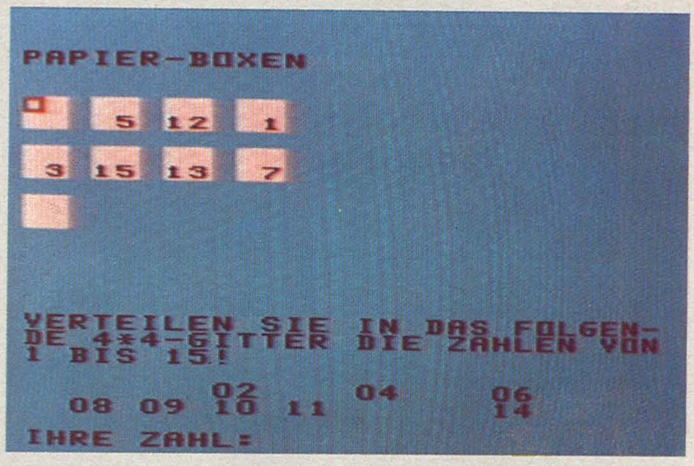
Bei einem neuen Spiel kann man mit gleicher Zahlenverteilung spielen, mit dem Computer die Felder tauschen oder mit neuer Zahlenverteilung spielen. Der Computer setzt seine Zahlen zufällig und nicht nach taktischen Gesichtspunkten, spielt aber sonst etwa mittelstark. Das Programm antwortet in Sekunden; falsche Züge werden von ihm nicht angenommen.

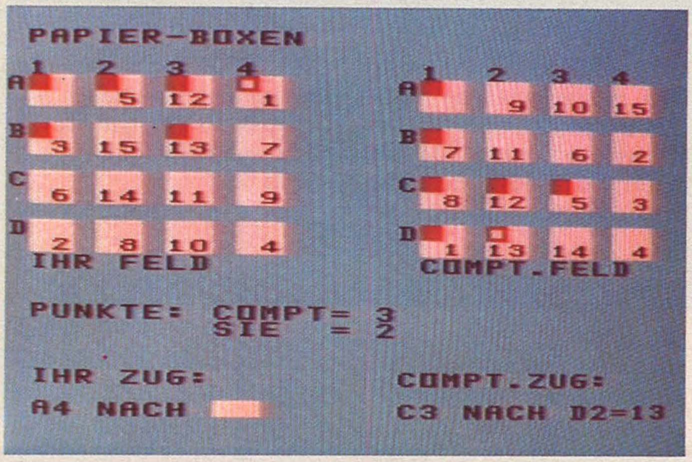
Zum Programm

Das Listing ist in Extended-BASIC geschrieben. Deshalb sind Leerstellen zwischen den Doppelpunkten der PRINT-Befehle, die TI-BASIC nicht braucht. Bei der Zahlenverteilung des Spielers müssen die Zahlen von 1 bis 9 als 01, 02 und so weiter eingegeben werden. Die einzelnen Zeilen:

30– 210: Initialisierung der Ausgangswerte
230– 290: Grundbewertung der Kästchen einlesen
310– 440: Sonderzeichen definieren und färben
450– 480: Titel „Papier-Boxen"
490– 660: Spielerfeld mit alten Zahlen malen
670– 700: Zahlenvorrat 1 bis 15 in Feld A() laden
720– 820: Zahlenvorratsreihe schreiben
840–1230: Zahlen als Vorrat wählen, prüfen, ob verfügbar, in Spielerfeld schreiben und aus Vorrat löschen
1240–1280: Text löschen, Spielerfeld-Koordinaten schreiben
1300–1600: Zahlenvorrat 1 bis 15 in Feld A() einlesen, daraus zufällig Zahl wählen, Computer-Feld mit Zahlen malen
1610–1700: Spielfelder und Punktanzeiger beschriften
1710–1790: Startspieler ermitteln
1810–2090: Spielerzug
2110–3040: Computerzug
2190: Wenn Computer vorgibt, dann weiter in 2230
2230–2290: Sucht höchste Zahl, die Spieler erreichen kann
2310–2660: Auswahlverfahren der möglichen Kästchen
2310–2350: Ausgangswerte für Kästchenwahl setzen

2370–2440: Sucht den höchsten Grundwert (MAXFLD) der noch betretbaren umliegenden Kästchen, um k.o.-Situation zu vermeiden
2450–2590: Wenn Kästchen schon betreten (C(I,J)=0), dann nächstes Kästchen untersuchen
2480: Wenn MAXFLD=−1, muß dieses Kästchen genommen werden (k.o. droht)
2500–2590: In den erlaubten Kästchen wird die absolut niedrigste Zahl gesucht (=MINZL), außerdem die Zahl, die genauso groß ist wie die dem Spieler mögliche Zahl (=MITZL), und die niedrigste Zahl, die gerade noch höher als die SZL ist (=MAXZL)
2600–2760: Wählt unter MINZL, MAXZL und MITZL aus
2600–2620: Gibt es eine MAXZL (<16), so wird dieses Kästchen gewählt
2630–2660: Gibt es dann eine MINZL (<16), so wird dieses Kästchen gewählt und zuletzt erst das gleichgroße MITZL-Kästchen. Die Zahl im gewählten Kästchen wird zum CWERT
2780–2950: Die Grundwerte der Kästchen um den CWERT-Kästchen werden um 1 erhöht
2960–3040: Gewähltes Kästchen schreiben, markieren
3060–3350: Punkte ermitteln und schreiben
3360: Ende-Test
3380–3530: Sind Spieler und Computer zugfähig? (k.o.-Test)
3550–4330: Endbehandlung
4340–4710: Modalitäten für neues Spiel festlegen
4730–4780: Schreibroutine ohne Scrolling
4800–4920: Routine, um Koordination umliegender Kästchen zu ermitteln (Cpt-Feld)
4940–5060: Routine, um Koordination umliegender Kästchen zu ermitteln (Spt-Feld)
5080–5100: Verzögerungsschleife





Die Variablen

FELD(): Grundbewertung der Kästchen im Computer-Feld
A(): Feld für Zahlenmenge von 1 bis 15
S(,): Feld für Zahlenverteilung im Spielerfeld
S1(,): Sicherung für S(,)
C(,): Feld für Zahlenverteilung im Computer-Feld
C1(,): Sicherung für C(,)
A$: Temporärer Zwischenspeicher
B$: Flag für Zahlenverteilung beim Spieler (= "J", wenn Zahlen neu verteilt werden sollen)
C$: Flag für Zahlenverteilung beim Computer (= "J", wenn Zahlen neu verteilt werden sollen)
D$: Flag für Tausch der Spielfelder, wenn "J"
M$: Text für Schreibroutine
X: Startspalte in Schreibroutine
Y: Zeile in Schreibroutine
W1: Charakterzähler in Schreibroutine
W2: ASCII-Code des zu schreibenden Charakters
Zug: Zähler für Spielzug
B: Flag (1=Spieler, 2=Computer zieht)
VG: Flag für Erst- und Zweitziehenden
I: Schleifenzähler
J: Schleifenzähler
Z: Temporärer Zwischenspeicher
K: Temporärer Zwischenspeicher und ähnliches
K, K1: Variable für Nachbarkästchen
K2, K3: Variable für Nachbarkästchen
ST: Statusvariable in Key-Routine
SZL: Zahl, die Spieler wählen kann
SWERT: Zahl, die Spieler gewählt hat
K, L: Buchstabenkoordinate des gewählten Spielerkästchens
K1, L1: Zahlkoordinate des gewählten Spielerkästchens
MAXFLD: Höchste Grundbewertung der zur Auswahl stehenden Kästchen
MINZL: Die niedrigste im Computer-Feld erreichbare Zahl
MITZL: Gleichhohe Zahl wie Spieler
MAXZL: Von den im Computer-Feld erreichbaren Zahlen die niedrigste, die aber höher als die Spielerzahl ist
CZL: Zahl, die Computer wählen kann
CWERT: Zahl, die Computer gewählt hat
N, M: Buchstabenkoordinate des gewählten Computer-Kästchens
N1, M1: Zahlkoordinate des gewählten Computer-Kästchens
CPKT: Zähler Computer-Punkte
SPKT: Zähler Spielerpunkte
CKO: Flag (=1, wenn Computer k.o.)
SKO: Flag (=1, wenn Spieler k.o.)

Bernhard Decker

```
10 REM  ********************
*       *    PAPIER-BOXEN
        * COPYRIGHT 12/1
981          * BY BERNHARD DE
CKER
20 REM  * TI 99/4A  TI-BASIC
        *    BRETTSPIEL VON
        *    SID SACKSON
          ****************
*****
30 CALL CLEAR
40 CALL SCREEN(12)
50 PRINT TAB(9);"PAPIER-BOXE
N"::::::::::::
60 OPTION BASE 1
70 DIM A(15),S(4,4),C(4,4),F
ELD(4,4),C1(4,4),S1(4,4)
80 B$="J"
90 C$="J"
100 D$="N"
110 RANDOMIZE
120 L=1
130 L1=1
140 N=1
```

```
150 N1=1
160 ZUG=1
170 VG=0
180 SPKT=0
190 CPKT=0
200 SKO=0
210 CKO=0
220 REM
230 RESTORE 290
240 FOR I=1 TO 4
250 FOR J=1 TO 4
260 READ FELD(I,J)
270 NEXT J
280 NEXT I
290 DATA 0,-3,-5,-2,-3,-7,-8
,-4,-4,-8,-8,-4,-2,-5,-5,-2
300 REM
310 RESTORE 400
320 FOR I=0 TO 9
330 READ A$
340 CALL CHAR(96+I,A$)
350 NEXT I
360 CALL CHAR(112,"0")
370 CALL CHAR(113,"FFFFFFFFF
FFFFFFF")
380 CALL CHAR(114,"FFFFC3C3C
3C3FFFF")
390 CALL CHAR(115,"103070FFF
F70301")
400 DATA 0038444444444438,0010
3010101038,0038440810207C,00
384418044438,00081828487
C08,0078407804443B
410 DATA 0038407844443B,007C
0408102020,0038443B444438,00
3844443C0478
420 CALL COLOR(9,2,16)
430 CALL COLOR(10,2,16)
440 CALL COLOR(11,9,16)
450 CALL CLEAR
460 CALL SCREEN(8)
470 PRINT "PAPIER-BOXEN"::::
::::::::::::
480 IF B$="J" THEN 670
490 PRINT :::::::
500 FOR I=1 TO 4
510 FOR J=1 TO 4
520 IF (I>1)+(J>1)<0 THEN 57
0
530 CALL HCHAR(4,3,114)
540 CALL HCHAR(4,4,112)
550 CALL HCHAR(5,3,112,2)
560 GOTO 640
570 CALL HCHAR(1+3*I,3*J,112
,2)
580 CALL HCHAR(2+3*I,3*J,112
,2)
590 Z=S(I,J)
600 IF Z<10 THEN 630
610 CALL HCHAR(2+3*I,3*J,97)
620 Z=Z-10
630 CALL HCHAR(2+3*I,1+3*J,9
6+Z)
640 NEXT J
650 NEXT I
660 GOTO 1250
670 FOR I=1 TO 15
680 A(I)=I
690 NEXT I
700 PRINT "VERTEILEN SIE IN
DAS FOLGEN-DE 4*4-GITTER DIE
 ZAHLEN VON1 BIS 15!":::
::
710 REM  ZAHLEN-PRINT
720 FOR I=0 TO 7
730 IF I=0 THEN 760
740 CALL HCHAR(21,5+3*I,48)
750 CALL HCHAR(21,6+3*I,48+A
(I))
760 IF A(I+8)<10 THEN 800
770 CALL HCHAR(22,5+3*I,49)
780 CALL HCHAR(22,6+3*I,38+A
(I+8))
790 GOTO 820
800 CALL HCHAR(22,5+3*I,48)
810 CALL HCHAR(22,6+3*I,48+A
(I+8))
820 NEXT I
830 REM  GITTER  SPIELER
840 M$="IHRE ZAHL:"
850 Y=24
860 GOSUB 4730
870 FOR I=1 TO 4
880 FOR J=1 TO 4
890 IF (I>1)+(J>1)<0 THEN 96
0
900 CALL HCHAR(4,3,114)
910 CALL HCHAR(4,4,112)
920 CALL HCHAR(5,3,112,2)
930 S(1,1)=0
940 S1(1,1)=0
950 GOTO 1210
960 CALL HCHAR(1+3*I,3*J,112
,2)
970 CALL HCHAR(2+3*I,3*J,112
,2)
980 CALL KEY(0,K,ST)
990 IF (ST<1)+(K<48)+(K>49)<
0 THEN 980
1000 CALL HCHAR(24,14,K)
1010 CALL KEY(0,M,ST)
1020 IF (ST<1)+(M<48)+(M>57)
<0 THEN 1010
1030 CALL HCHAR(24,15,M)
1040 Z=(K-48)*10+M-48
1050 IF (Z<1)+(Z>15)<0 THEN
1070
1060 IF A(Z)>0 THEN 1090
1070 CALL HCHAR(24,14,32,2)
1080 GOTO 980
1090 A(Z)=0
1100 S(I,J)=Z
1110 S1(I,J)=Z
1120 IF Z>7 THEN 1150
1130 CALL HCHAR(21,Z*3+5,32,
2)
1140 GOTO 1160
1150 CALL HCHAR(22,(Z-8)*3+5
,32,2)
1160 CALL HCHAR(1+3*I,3*J,11
2,2)
1170 IF Z<10 THEN 1200
1180 CALL HCHAR(2+3*I,3*J,97
)
1190 Z=Z-10
1200 CALL HCHAR(2+3*I,1+3*J,
96+Z)
1210 CALL HCHAR(24,14,32,2)
1220 NEXT J
1230 NEXT I
1240 CALL HCHAR(17,1,32,256)
1250 FOR I=1 TO 4
1260 CALL HCHAR(1+3*I,2,64+I
)
1270 CALL HCHAR(3,3*I,48+I)
1280 NEXT I
1290 REM  MISCHEN+GITTER COM
PT
1300 IF C$="N" THEN 1340
1310 FOR I=1 TO 15
1320 A(I)=1
1330 NEXT I
1340 FOR I=1 TO 4
1350 FOR J=1 TO 4
1360 IF (I>1)+(J>1)<0 THEN 1
410
1370 CALL HCHAR(1+3*I,17+3*J
,114)
1380 CALL HCHAR(1+3*I,18+3*J
,112)
1390 CALL HCHAR(2+3*I,17+3*J
,112,2)
1400 GOTO 1550
1410 CALL HCHAR(1+3*I,17+3*J
,112,2)
1420 CALL HCHAR(2+3*I,17+3*J
,112,2)
1430 IF C$="J" THEN 1460
1440 Z=C(I,J)
1450 GOTO 1510
1460 Z=INT(RND*15)+1
1470 IF A(Z)=0 THEN 1460
1480 A(Z)=0
1490 C(I,J)=Z
1500 C1(I,J)=Z
1510 IF Z<10 THEN 1540
1520 CALL HCHAR(2+3*I,17+3*J
,97)
1530 Z=Z-10
1540 CALL HCHAR(2+3*I,18+3*J
,96+Z)
1550 NEXT J
1560 NEXT I
1570 FOR I=1 TO 4
1580 CALL HCHAR(1+3*I,19,64+
I)
1590 CALL HCHAR(3,17+3*I,48+
I)
1600 NEXT I
1610 M$="IHR FELD          CO
MPT.FELD"
1620 Y=15
1630 GOSUB 4730
1640 M$="PUNKTE: COMPT="
1650 Y=18
1660 GOSUB 4740
1670 M$="SIE   ="
1680 Y=19
1690 X=10
1700 GOSUB 4740
1710 Z=S(4,4)+C(4,4)
1720 IF Z/2=INT(Z/2)THEN 176
0
1730 IF D$="J" THEN 1770
1740 B=1
1750 GOTO 1810
1760 IF D$="J" THEN 1740
1770 B=2
1780 GOTO 2110
1790 REM
1800 REM  SPIELERZUG
1810 M$="IHR ZUG:"
1820 Y=22
1830 GOSUB 4730
1840 M$=CHR$(64+L)&CHR$(48+L
1)&" NACH"
1850 Y=24
1860 GOSUB 4740
1870 CALL SOUND(100,800,6)
1880 CALL HCHAR(24,11,112,2)
1890 CALL KEY(0,K,ST)
1900 IF (K<65)+(K>68)<0 THEN
```

Scan: S. Höltgen

```
1890
1910 CALL HCHAR(24,11,K)
1920 CALL KEY(0,K1,ST)
1930 IF (K1<49)+(K1>52)<0 TH
EN 1920
1940 CALL HCHAR(24,12,K1)
1950 K=K-64
1960 K1=K1-48
1970 IF S(K,K1)=0 THEN 1850
1980 IF (ABS(K-L)>1)+(ABS(K1
-L1)>1)<0 THEN 1850
1990 SWERT=S(K,K1)
2000 S(K,K1)=0
2010 VG=VG+1
2020 CALL HCHAR(1+3*L,3*L1,1
13)
2030 CALL HCHAR(1+3*K,3*K1,1
14)
2040 L=K
2050 L1=K1
2060 M$="="&STR$(SWERT)
2070 X=12
2080 GOSUB 4740
2090 ON VG GOTO 2110,3060
2100 REM
2110 M$="COMPT.ZUG:"
2120 Y=22
2130 X=18
2140 GOSUB 4740
2150 M$=CHR$(64+N)&CHR$(48+N
1)&" NACH"
2160 Y=24
2170 GOSUB 4740
2180 CALL HCHAR(24,27,112,2)
2190 IF VG<>1 THEN 2230
2200 SZL=SWERT
2210 GOTO 2310
2220 REM  NAECHSTHOECHSTE SP
IELERZAHL
2230 GOSUB 4940
2240 FOR I=K TO K1
2250 FOR J=K2 TO K3
2260 IF SZL>=S(I,J)THEN 2280
2270 SZL=S(I,J)
2280 NEXT J
2290 NEXT I
2300 REM  NIEDRIGST COMPTZAH
L,GROESSER ALS SZAHL
2310 Z=1
2320 MITZL=-1
2330 MINZL=16
2340 MAXZL=16
2350 MAXFLD=-10
2360 REM  MAX SETZFELDWERT
2370 GOSUB 4800
2380 FOR I=K TO K1
2390 FOR J=K2 TO K3
2400 IF C(I,J)=0 THEN 2430
2410 IF MAXFLD>FELD(I,J)THEN
 2430
2420 MAXFLD=FELD(I,J)
2430 NEXT J
2440 NEXT I
2450 FOR I=K TO K1
2460 FOR J=K2 TO K3
2470 IF C(I,J)=0 THEN 2580
2480 IF MAXFLD<-1 THEN 2500
2490 IF FELD(I,J)<-1 THEN 25
80
2500 IF C(I,J)-SZL>-1 THEN 2
530
2510 IF MINZL<C(I,J)THEN 253
0
2520 MINZL=C(I,J)
2530 IF C(I,J)-SZL<1 THEN 25
60
2540 IF MAXZL<C(I,J)THEN 256
0
2550 MAXZL=C(I,J)
2560 IF C(I,J)<>SZL THEN 258
0
2570 MITZL=C(I,J)
2580 NEXT J
2590 NEXT I
2600 IF MAXZL>15 THEN 2630
2610 CZL=MAXZL
2620 GOTO 2670
2630 IF MINZL<16 THEN 2660
2640 CZL=MITZL
2650 GOTO 2670
2660 CZL=MINZL
2670 FOR I=1 TO 4
2680 FOR J=1 TO 4
2690 IF C(I,J)=CZL THEN 2720
2700 NEXT J
2710 NEXT I
2720 M=I
2730 M1=J
2740 CWERT=C(M,M1)
2750 C(M,M1)=0
2760 VG=VG+1
2770 REM  UPDATE FELDWERTE U
M SETZFELD
2780 K=M-1
2790 IF K>=1 THEN 2810
2800 K=1
2810 K1=M+1
2820 IF K1<=4 THEN 2840
2830 K1=4
2840 K2=M1-1
2850 IF K2>=1 THEN 2870
2860 K2=1
2870 K3=M1+1
2880 IF K3<=4 THEN 2900
2890 K3=4
2900 FOR I=K TO K1
2910 FOR J=K2 TO K3
2920 IF C(I,J)=0 THEN 2940
2930 FELD(I,J)=FELD(I,J)+1
2940 NEXT J
2950 NEXT I
2960 M$=CHR$(64+M)&CHR$(48+M
1)&"="&STR$(CWERT)
2970 Y=24
2980 X=26
2990 GOSUB 4740
3000 CALL HCHAR(1+3*N,17+3*N
1,113)
3010 CALL HCHAR(1+3*M,17+3*M
1,114)
3020 N=M
3030 N1=M1
3040 ON VG GOTO 1810,3060
3050 REM  PUNKTE
3060 VG=0
3070 IF CWERT>=SWERT THEN 31
60
3080 SPKT=SPKT+1
3090 B=1
3100 CALL SOUND(100,600,4)
3110 FOR I=1 TO 6
3120 CALL HCHAR(19,20,115)
3130 CALL HCHAR(19,20,32)
3140 NEXT I
3150 GOTO 3270
3160 IF CWERT=SWERT THEN 325
0
3170 CPKT=CPKT+1
```


DAS COMPUTER PROGRAMM

Der Alphacom 32

Thermopapier Printer mit eingebautem Interface für den Sinclair ZX81 und ZX Spectrum. Ein ungewöhnlich robuster Rollenpapier-Printer, kompakt und handlich, extrem problemlos in der Anwendung, unermüdlich funktionstüchtig dank eingebauter Ventilation. Energieversorgung durch separaten Power-Adaptor. Einfacher Steckanschluß an Ihren Computer.

Und das bietet der Alphacom 32: Durch einfachen Steckeranschluß sofort betriebsbereit. Eingebautes Interface für den Sinclair ZX81 und ZXSpectrum, mit allen Befehlen voll Sinclair-kompatibel. 32 Zeichen pro Zeile. Ausdruck aller Grafikzeichen. Hochauflösende Grafik. Sehr klarer Ausdruck auf Thermopapier. Druckt 2 Zeilen pro Sekunde.
Lieferumfang: Drucker mit eingebautem Interface, Steckanschlüsse, eine Rolle Thermopapier, Power-Adaptor und ausführliche Beschreibung.
Nr. 106,

DM 298,–

COMPUTER ACCESSOIRES INT'L
Der Computer-Ausstatter.

Hier wird bestellt:

☐ per Vorausscheck
☐ per Nachnahme (zuzügl. Nachnahmegeb.)

Stück	Alphacom 32 Thermopapier Printer Nr. 106	Einzel-Preis 298,–	Gesamt-Preis

Name ____
Straße ____
PLZ/Ort ____
Datum ____
Unterschrift ____
Bei Bestellungen unter DM 250,– zuzügl. Versandspesen.

COMPUTER ACCESSOIRES INT'L GMBH
Jägerweg 10 – 8012 Ottobrunn HC

Scan: S. Höltgen


```
3180 B=2
3190 CALL SOUND(100,300,4)
3200 FOR I=1 TO 6
3210 CALL HCHAR(18,20,115)
3220 CALL HCHAR(18,20,32)
3230 NEXT I
3240 GOTO 3270
3250 CALL SOUND(100,600,4)
3260 CALL SOUND(100,300,4)
3270 M$=STR$(CPKT)
3280 Y=18
3290 X=17
3300 GOSUB 4740
3310 M$=STR$(SPKT)
3320 Y=19
3330 GOSUB 4740
3340 ZUG=ZUG+1
3350 GOSUB 5080
3360 IF ZUG=16 THEN 3550
3370 GOSUB 4940
3380 FOR I=K TO K1
3390 FOR J=K2 TO K3
3400 IF S(I,J)>0 THEN 3440
3410 NEXT J
3420 NEXT I
3430 SKO=1
3440 GOSUB 4800
3450 FOR I=K TO K1
3460 FOR J=K2 TO K3
3470 IF C(I,J)>0 THEN 3510
3480 NEXT J
3490 NEXT I
3500 CKO=1
3510 CALL HCHAR(20,1,32,160)
3520 IF (CKO=1)+(SKO=1)<0 TH
EN 3550
3530 ON B GOTO 1810,2110
3540 REM  ENDE
3550 CALL HCHAR(20,1,32,160)
3560 IF (SKO=1)+(CKO=1)<0 TH
EN 4080
3570 IF CPKT>=SPKT THEN 3900
3580 FOR I=1 TO 5
3590 CALL SOUND(-600,400*I/2
,3,600*I/3,5,800*I/5,7)
3600 NEXT I
3610 M$="BRAVO!"
3620 Y=23
3630 GOSUB 4730
3640 M$="SIE HABEN GEWONNEN.
"
3650 Y=24
3660 GOSUB 4740
3670 IF (SPKT-CPKT>1)+(D$="J
")<0 THEN 4340
3680 GOSUB 5080
3690 CALL HCHAR(23,1,32,64)
3700 CALL SOUND(100,1000,4)
3710 CALL SOUND(400,800,4)
3720 M$="DER COMPUTER FORDER
T"
3730 Y=20
3740 GOSUB 4730
3750 M$="REVANCHE!!"
3760 Y=21
3770 GOSUB 4740
3780 M$="DAS NAECHSTE SPIEL
WIRD MIT"
3790 Y=23
3800 GOSUB 4740
3810 M$="GETAUSCHTEN ZAHLEN
GESPIELT."
3820 Y=24
3830 GOSUB 4740
3840 GOSUB 5080
3850 D$="J"
3860 B$="N"
3870 C$="N"
3880 GOTO 4650
3890 REM  COMPT GEWINNT
3900 IF CPKT=SPKT THEN 4020
3910 FOR I=5 TO 1 STEP -1
3920 CALL SOUND(-600,400*I/2
,3,600*I/3,5,800*I/5,7)
3930 NEXT I
3940 M$="NANU, NANU!"
3950 Y=23
3960 GOSUB 4730
3970 M$="SIE HABEN JA VERLOR
EN!!!"
3980 Y=24
3990 GOSUB 4740
4000 GOTO 4340
4010 REM  REMIS
4020 CALL SOUND(700,400,3,60
0,4,800,5)
4030 M$="UNENTSCHIEDEN!"
4040 Y=24
4050 GOSUB 4730
4060 GOTO 4340
4070 REM  KO
4080 IF SKO=1 THEN 4170
4090 CALL SOUND(600,800,3,10
00,3,1150,7)
4100 M$="COMPUTER KANN NICHT
 ZIEHEN."
4110 Y=21
4120 GOSUB 4730
4130 M$="SIE GEWINNEN DURCH
K.O.!!"
4140 Y=24
4150 GOSUB 4740
4160 GOTO 4340
4170 IF CKO=1 THEN 4260
4180 CALL SOUND(900,900,3,92
0,4,950,5)
4190 M$="SIE KOENNEN NICHT Z
IEHEN."
4200 Y=21
4210 GOSUB 4730
4220 M$="COMPUTER GEWINNT DU
RCH K.O.!"
4230 Y=24
4240 GOSUB 4740
4250 GOTO 4340
4260 CALL SOUND(600,800,3,10
00,3,1150,7)
4270 CALL SOUND(900,900,3,92
0,4,950,5)
4280 M$="KEINER KANN MEHR ZI
EHEN."
4290 Y=20
4300 GOSUB 4730
4310 M$="UNENTSCHIEDEN!!!"
4320 Y=24
4330 GOSUB 4740
4340 CALL KEY(0,I,ST)
4350 IF ST=0 THEN 4340
4360 CALL CLEAR
4370 INPUT "NEUES SPIEL? (J/
N): ":A$
4380 IF A$="J" THEN 4400
4390 STOP
4400 PRINT ::"WOLLEN SIE IHR
E ZAHLEN NEU"
4410 INPUT "VERTEILEN? (J/N)
: ":B$
4420 IF B$="J" THEN 4490
4430 IF B$<>"N" THEN 4400
4440 FOR I=1 TO 4
4450 FOR J=1 TO 4
4460 S(I,J)=S1(I,J)
4470 NEXT J
4480 NEXT I
4490 PRINT ::"SOLLEN DIE COM
PUTERZAHLEN":"NEU VERTEILT W
ERDEN?"
4500 INPUT "(J/N): ":C$
4510 IF C$="J" THEN 4580
4520 IF C$<>"N" THEN 4490
4530 FOR I=1 TO 4
4540 FOR J=1 TO 4
4550 C(I,J)=C1(I,J)
4560 NEXT J
4570 NEXT I
4580 IF (B$="N")+(C$="N")=-2
 THEN 4610
4590 D$="N"
4600 GOTO 120
4610 PRINT ::"WOLLEN SIE MIT
 DEM COMPUTER DIE ZAHLEN TAU
SCHEN?"
4620 INPUT "(J/N): ":D$
4630 IF D$="N" THEN 4710
4640 IF D$<>"J" THEN 4610
4650 FOR I=1 TO 4
4660 FOR J=1 TO 4
4670 S(I,J)=C1(I,J)
4680 C(I,J)=S1(I,J)
4690 NEXT J
4700 NEXT I
4710 GOTO 120
4720 REM  SCHREIBEN
4730 X=2
4740 FOR W1=1 TO LEN(M$)
4750 W2=ASC(SEG$(M$,W1,1))
4760 CALL HCHAR(Y,X+W1,W2)
4770 NEXT W1
4780 RETURN
4790 REM
4800 K=N-1
4810 IF K>=1 THEN 4830
4820 K=1
4830 K1=N+1
4840 IF K1<=4 THEN 4860
4850 K1=4
4860 K2=N1-1
4870 IF K2>=1 THEN 4890
4880 K2=1
4890 K3=N1+1
4900 IF K3<=4 THEN 4920
4910 K3=4
4920 RETURN
4930 REM
4940 K=L-1
4950 IF K>=1 THEN 4970
4960 K=1
4970 K1=L+1
4980 IF K1<=4 THEN 4990
4990 K1=4
5000 K2=L1-1
5010 IF K2>=1 THEN 5030
5020 K2=1
5030 K3=L1+1
5040 IF K3<=4 THEN 5060
5050 K3=4
5060 RETURN
5070 REM
5080 FOR I=1 TO 400
5090 NEXT I
5100 RETURN
```

Scan: S. Höltgen

Skatschreiber

Eine deftige Skatrunde, und kein Spieler muß den aktuellen Punktestand berechnen. Diese Aufgabe übernimmt der TI-99/4A in der Grundversion unter minimaler Belastung der Spieler

Die Spieler sollen sich so weit wie möglich auf die eigentlichen Spiele konzentrieren können. So gibt der Computer zum Beispiel bei jeder Runde den Namen des jeweiligen Gebers an, zeigt die Anzahl von Bockrunden an und berechnet dementsprechend, wie viele Spiele jeder Spieler gemacht hat und wie viele Spiele insgesamt gespielt wurden. Falsche Eingaben können korrigiert werden.

Populäre Regeln

Bei Nullspielen entfallen die Abfragen „Zählwert" bis „Schwarz angesagt". Der Computer fragt also nur solche Angaben ab, die für das jeweilige Spiel relevant sind und den Skatregeln entsprechen. Regelwidrige Eingaben werden somit vermieden.

Der Computer rechnet nach den allgemein bekannten und populären Skatregeln. Darunter ist das Reizen auch mit „Kontra", „Re" bis „Noch ein Drupp" zu verstehen.

Anpassung für andere Rechner

Die Besitzer anderer Home-Computer können das Programm ohne große Probleme übernehmen, da die meisten Befehle Standard bei fast allen Home-Computern sind. Computerspezifische Befehle werden in REMs erklärt. Auch beim Abändern der Variablen für Computer, die nicht mehr als zwei Stellen zulassen, sollten keine Schwierigkeiten entstehen.

Alles Wissenswerte zur Nutzung wird in einem Anmerkungsblock innerhalb des Programms erklärt.

Hans-Jürgen Sawinski

```
100 REM
110 REM  skat-aufschreiber
120 REM
130 REM
140 REM  programm von
150 REM
160 REM  hans-juergen
170 REM
180 REM    sawinski
190 REM
200 REM   dornstr. 3
210 REM
220 REM  4690 herne 1
230 REM
240 REM    telefon:
250 REM
260 REM  02323/51532
270 REM
280 REM
290 REM
300 REM  call clear = bildsc
hirm loseschen
310 REM
320 REM   call key = eingabe
 ueber tastatur
330 REM
340 REM
350 CALL CLEAR
360 PRINT "       *** ANMERKUN
G ***"
370 PRINT "       ===========
=====" 
380 PRINT :::
390 PRINT "BEI SPIELERN MIT
GLEICHEM    NAMEN, MUESSEN DI
E NAMEN     ABGEAENDERT W
ERDEN Z. B."
400 PRINT "2*HUGO DANN HUGO1
 UND HUGO2.":::
410 PRINT "ZEICHENERKLAERUNG
:"::
420 PRINT ">> = WEITER MIT L
EERTASTE":::
430 PRINT " a = NEUEBEGINN S
KAT"::
440 PRINT " e = PROGRAMMMODU
S VERLASSEN"::
450 PRINT "
          >>"
460 CALL KEY(0,KEY,STATUS)
470 IF KEY<>32 THEN 460
480 CALL CLEAR
490 PRINT "ANMERKUNG ZUR SPI
ELABFRAGE :"
500 PRINT "=================
===========":: 
510 PRINT "ALPHA-LOCK-TASTE
DRUCKEN, DASPIELWERTE IN GRO
SS-        SCHRIFT GESPE
ICHERT SIND!"::
520 PRINT "SPIELWERTE MUESSE
N WIE FOLGTEINGEGEBEN WERTEN
: KREUZ,     PIK, HERZ, KA
RO, GRAND, "
530 PRINT "GRANDOVERT, NULL,
 NULLOVERT,REVOLUTION, NULLH
AND, NULL-  OVERTHAND!"::
540 PRINT "BOCKRUNDE = ZEIGT
 AN WIEVIELBOCKRUNDEN LAUFEN
. BEI BOCK-SPIELEN ERREC
HNET DER COMPU-"
550 PRINT "DIE ENTSPRECHEND
HOEHER AUS-FALLENDE PUNKTZAH
L AUTOMA-   TISCH."::
560 PRINT "
          >>"
570 CALL KEY(0,KEY,STATUS)
580 IF KEY<>32 THEN 570
590 CALL CLEAR
600 PRINT "3 BOCKRUNDEN? = A
M ENDE VON JEDEM SPIEL, FRAG
T DER COM-  PUTER, OB DIE
 NAECHSTEN 3"
610 PRINT "RUNDEN BOCKRUNDEN
 SIND.      DER RECHNER ZAEHL
T AUCH GGF.NEUE BOCKRUND
EN ZU DEN ALTEN"::
620 PRINT "ZAEHLWERT=MULTIPL
IKATOR AUF-GRUND DER ANZAHL
VON BAUERN.DIE ZAEHLER K
OENNEN VON 2"
630 PRINT "BIS 11 REICHEN(MI
T 1 SPIEL   2 BIS OHNE 10 SPI
EL 11)."::
640 PRINT "ANSONSTEN GELTEN
DIE ALLGE-  MEIN BEKANNTEN SK
AT-REGELN.":::
650 PRINT "
          >>"
660 CALL KEY(0,KEY,STATUS)
670 IF KEY<>32 THEN 660
680 GEBER=0
690 SPIELWERT1=0
700 SPIELWERT2=0
710 SPIELWERT3=0
720 SPIELWERT4=0
730 ANZAHL1=0
740 ANZAHL2=0
750 ANZAHL3=0
760 ANZAHL4=0
770 TEILZAHL=0
780 BOCKWERT=0
790 BOCKWERT1=4
800 CALL CLEAR
810 PRINT "S K A T - AUFSCHR
EIBER ! ! !"
820 PRINT "=================
==========="
830 PRINT :::
840 INPUT "WIEVIELE SPIELER(
3 O. 4)? ":TEILZAHL
850 PRINT "-----------------
-----------"
860 IF TEILZAHL<3 THEN 840
870 IF TEILZAHL>4 THEN 840
880 PRINT ::
890 INPUT "1. SPIELER: ":SPI
ELER1$
900 IF SPIELER1$="" THEN 890
910 PRINT
920 INPUT "2. SPIELER: ":SPI
ELER2$
930 IF SPIELER2$="" THEN 920
940 PRINT
950 INPUT "3. SPIELER: ":SPI
ELER3$
960 IF SPIELER3$="" THEN 950
970 IF TEILZAHL=4 THEN 980 E
```

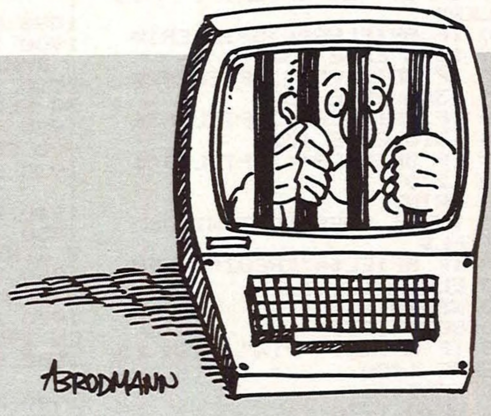
Scan: S. Höltgen


```
LSE 1010
980 PRINT
990 INPUT "4. SPIELER: ":SPI
ELER4$
1000 IF SPIELER4$="" THEN 99
0
1010 PRINT ::
1020 PRINT "----------------
-----------"
1030 INPUT "EINGABEN OKAY?
J/N: ":OK$
1040 IF OK$="J" THEN 1070 EL
SE 1050
1050 IF OK$<>"N" THEN 1030
1060 GOTO 680
1070 SPIELE=1
1080 CALL CLEAR
1090 GOTO 1170
1100 SPIEL$=""
1110 WERT=0
1120 BAUERN=0
1130 SPIELWERT=0
1140 GEBER=GEBER-1
1150 BOCKWERT=BOCKWERT+1
1160 CALL CLEAR
1170 PRINT "S K A T    -    R
U N D E ";SPIELE
1180 PRINT "================
============"
1190 PRINT ::
1200 GOSUB 3070
1210 PRINT "GEBER IST :    ";G
EBER$
1220 PRINT
1230 IF BOCKWERT>0 THEN 1240
 ELSE 1250
1240 BOCKWERT=BOCKWERT-1
1250 PRINT "BOCKRUNDEN: ";BO
CKWERT
1260 PRINT
1270 INPUT "ES SPIELT :    ":S
PIELER$
1280 IF SPIELER$<>SPIELER1$
THEN 1290 ELSE 1320
1290 IF SPIELER$<>SPIELER2$
THEN 1300 ELSE 1320
1300 IF SPIELER$<>SPIELER3$
THEN 1310 ELSE 1320
1310 IF SPIELER$<>SPIELER4$
THEN 1270
1320 PRINT
1330 INPUT "GESPIELT WIRD:
":SPIEL$
1340 IF SPIEL$="KREUZ" THEN
1350 ELSE 1370
1350 WERT=12
1360 GOTO 1730
1370 IF SPIEL$="PIK" THEN 13
80 ELSE 1400
1380 WERT=11
1390 GOTO 1730
1400 IF SPIEL$="HERZ" THEN 1
410 ELSE 1430
1410 WERT=10
1420 GOTO 1730
1430 IF SPIEL$="KARO" THEN 1
440 ELSE 1460
1440 WERT=9
1450 GOTO 1730
1460 IF SPIEL$="GRAND" THEN
1470 ELSE 1490
1470 WERT=24
1480 GOTO 1730
1490 IF SPIEL$="GRANDOVERT"
THEN 1500 ELSE 1520
1500 WERT=36
1510 GOTO 1730
1520 IF SPIEL$="NULL" THEN 1
530 ELSE 1560
1530 WERT=23
1540 SPIELWERT=23
1550 GOTO 2040
1560 IF SPIEL$="NULLOVERT" T
HEN 1570 ELSE 1600
1570 WERT=46
1580 SPIELWERT=46
1590 GOTO 2040
1600 IF SPIEL$="REVOLUTION"
THEN 1610 ELSE 1640
1610 WERT=92
1620 SPIELWERT=92
1630 GOTO 2040
1640 IF SPIEL$="NULLHAND" TH
EN 1650 ELSE 1680
1650 WERT=36
1660 SPIELWERT=36
1670 GOTO 2040
1680 IF SPIEL$="NULLOVERTHAN
D" THEN 1690 ELSE 1720
1690 WERT=59
1700 SPIELWERT=59
1710 GOTO 2040
1720 GOTO 1330
1730 PRINT
1740 INPUT "ZAEHLWERT?    ":B
AUERN
1750 GOTO 1780
1760 IF BAUERN<2 THEN 1740
1770 IF BAUERN>11 THEN 1740
1780 SPIELWERT=WERT*BAUERN
1790 PRINT
1800 INPUT "HAND?
 J/N: ":HAND$
1810 IF HAND$="J" THEN 1830
ELSE 1820
1820 IF HAND$<>"N" THEN 1800
 ELSE 1840
1830 SPIELWERT=SPIELWERT+WER
T
1840 PRINT
1850 INPUT "SCHNEIDER?
 J/N: ":SCHNEIDER$
1860 IF SCHNEIDER$="J" THEN
1880 ELSE 1870
1870 IF SCHNEIDER$<>"N" THEN
 1850 ELSE 1940
1880 SPIELWERT=SPIELWERT+WER
T
1890 PRINT
1900 INPUT "SCHWARZ?
 J/N: ":SCHWARZ$
1910 IF SCHWARZ$="J" THEN 19
30 ELSE 1920
1920 IF SCHWARZ$<>"N" THEN 1
900 ELSE 1940
1930 SPIELWERT=SPIELWERT+WER
T
1940 PRINT
1950 INPUT "SCHNEIDER ANGES.
? J/N: ":SCHNEIDERAN$
1960 IF SCHNEIDERAN$="J" THE
N 1980 ELSE 1970
1970 IF SCHNEIDERAN$<>"N" TH
EN 1950 ELSE 2040
1980 SPIELWERT=SPIELWERT+WER
T
1990 PRINT
2000 INPUT "SCHWARZ ANGES.?
 J/N: ":SCHWARZAN$
2010 IF SCHWARZAN$="J" THEN
2030 ELSE 2020
2020 IF SCHWARZAN$<>"N" THEN
 2000 ELSE 2040
2030 SPIELWERT=SPIELWERT+WER
T
2040 PRINT
2050 INPUT "KONTRA?
 J/N: ":KONTRA$
2060 IF KONTRA$="J" THEN 208
0 ELSE 2070
2070 IF KONTRA$<>"N" THEN 20
50 ELSE 2340
2080 SPIELWERT=SPIELWERT*2
2090 PRINT
2100 INPUT "RE?
 J/N: ":RE$
2110 IF RE$="J" THEN 2130 EL
SE 2120
2120 IF RE$<>"N" THEN 2100 E
LSE 2360
2130 SPIELWERT=SPIELWERT*2
2140 PRINT
2150 INPUT "BOCK?
 J/N: ":BOCK2$
2160 IF BOCK2$="J" THEN 2180
 ELSE 2170
2170 IF BOCK2$<>"N" THEN 215
0 ELSE 2340
2180 SPIELWERT=SPIELWERT*2
2190 PRINT
2200 INPUT "HIRSCH?
 J/N: ":HIRSCH$
2210 IF HIRSCH$="J" THEN 223
0 ELSE 2220
2220 IF HIRSCH$<>"N" THEN 22
00 ELSE 2340
2230 SPIELWERT=SPIELWERT*2
2240 PRINT
2250 INPUT "ZUPP?
 J/N: ":ZUPP$
2260 IF ZUPP$="J" THEN 2280
ELSE 2270
2270 IF ZUPP$<>"N" THEN 2250
 ELSE 2340
2280 SPIELWERT=SPIELWERT*2
2290 PRINT
2300 INPUT "NOCH EIN DRUPP?
 J/N: ":DRUPP$
2310 IF DRUPP$="J" THEN 2330
 ELSE 2320
2320 IF DRUPP$<>"N" THEN 230
0 ELSE 2340
2330 SPIELWERT=SPIELWERT*2
2340 IF BOCKWERT>0 THEN 2350
 ELSE 2360
2350 SPIELWERT=SPIELWERT*2
2360 PRINT
2370 INPUT "SPIEL GEWONNEN?
 J/N: ":SPIEL1$
2380 IF SPIEL1$="J" THEN 241
0 ELSE 2390
2390 IF SPIEL1$<>"N" THEN 23
70 ELSE 2400
2400 SPIELWERT=SPIELWERT*-2
2410 PRINT
2420 INPUT "3 BOCKRUNDEN?
 J/N: ":BOCK1$
2430 IF BOCK1$="J" THEN 2460
 ELSE 2440
2440 IF BOCK1$<>"N" THEN 242
0
2450 GOTO 2470
2460 GOSUB 3280
2470 PRINT
2480 INPUT "EINGABEN OKAY?
 J/N: ":OKAY$
2490 IF OKAY$="J" THEN 2530
2500 IF OKAY$<>"N" THEN 2480
2510 GOSUB 3320
2520 GOTO 1100
2530 PRINT
2540 IF SPIELER$=SPIELER1$ T
HEN 2550 ELSE 2570
2550 GOSUB 2950
2560 GOTO 2640
2570 IF SPIELER$=SPIELER2$ T
HEN 2580 ELSE 2600
2580 GOSUB 2980
2590 GOTO 2640
2600 IF SPIELER$=SPIELER3$ T
HEN 2610 ELSE 2630
2610 GOSUB 3010
2620 GOTO 2640
2630 GOSUB 3040
2640 CALL CLEAR
2650 PRINT "STAND    NACH    R
U N D E ";SPIELE
2660 PRINT "================
============"
2670 PRINT
2680 PRINT "ZULETZT GESPIELT
ES SPIEL:"
2690 PRINT
2700 PRINT "SPIELER         :
";SPIELER$
2710 PRINT "SPIEL           :
";SPIEL$
2720 PRINT "WERT            : "
;WERT
```

Scan: S. Höltgen


```
2730 PRINT "ZAEHLWERT     : "
;BAUERN
2740 PRINT "PUNKTE        : "
;SPIELWERT
2750 PRINT
2760 PRINT "NAME","PUNKTE";"
  SPIELE"
2770 PRINT "================
============"
2780 PRINT
2790 PRINT SPIELER1$,SPIELWE
RT1;TAB(22);ANZAHL1
2800 PRINT
2810 PRINT SPIELER2$,SPIELWE
RT2;TAB(22);ANZAHL2
2820 PRINT
2830 PRINT SPIELER3$,SPIELWE
RT3;TAB(22);ANZAHL3
2840 IF TEILZAHL=4 THEN 2850
 ELSE 2870
2850 PRINT
2860 PRINT SPIELER4$,SPIELWE
RT4;TAB(22);ANZAHL4
2870 PRINT
2880 PRINT "
      >>/a/e"
2890 CALL KEY(0,KEY,STATUS)
2900 IF KEY<>32 THEN 2910 EL
SE 2930
2910 IF KEY<>97 THEN 2920 EL
SE 100
2920 IF KEY<>101 THEN 2890 E
LSE 3370
2930 SPIELE=SPIELE+1
2940 GOTO 1080
2950 SPIELWERT1=SPIELWERT1+S
PIELWERT
2960 ANZAHL1=ANZAHL1+1
2970 RETURN
2980 SPIELWERT2=SPIELWERT2+S
PIELWERT
2990 ANZAHL2=ANZAHL2+1
3000 RETURN
3010 SPIELWERT3=SPIELWERT3+S
PIELWERT
3020 ANZAHL3=ANZAHL3+1
3030 RETURN
3040 SPIELWERT4=SPIELWERT4+S
PIELWERT
3050 ANZAHL4=ANZAHL4+1
3060 RETURN
3070 GEBER=GEBER+1
3080 IF GEBER=1 THEN 3090 EL
SE 3110
3090 GEBER$=SPIELER1$
3100 GOTO 3270
3110 IF GEBER=2 THEN 3120 EL
SE 3140
3120 GEBER$=SPIELER2$
3130 GOTO 3270
3140 IF GEBER=3 THEN 3150 EL
SE 3170
3150 GEBER$=SPIELER3$
3160 GOTO 3270
3170 IF TEILZAHL=4 THEN 3180
 ELSE 3240
3180 IF GEBER=4 THEN 3190 EL
SE 3210
3190 GEBER$=SPIELER4$
3200 GOTO 3270
3210 IF GEBER>5 THEN 3220
3220 GEBER=0
3230 GOTO 3070
3240 IF GEBER>4 THEN 3250
3250 GEBER=0
3260 GOTO 3070
3270 RETURN
3280 BOCKWERT=BOCKWERT+BOCKW
ERT1
3290 IF BOCKWERT>4 THEN 3300
 ELSE 3310
3300 BOCKWERT=BOCKWERT-1
3310 RETURN
3320 IF BOCK1$="J" THEN 3330
 ELSE 3360
3330 BOCKWERT=BOCKWERT-BOCKW
ERT1
3340 IF BOCKWERT>0 THEN 3350
 ELSE 3360
3350 BOCKWERT=BOCKWERT+1
3360 RETURN
3370 CALL CLEAR
3380 END
```

Knacki



Der Ausbruch aus dem Gefängnis ist geglückt. Das Ziel ist jedoch erst erreicht, wenn in insgesamt sieben Phasen die Schlüssel auf dem Bildschirm eingesammelt sind (Spectrum 48K, HC-Diplom-Aktion)

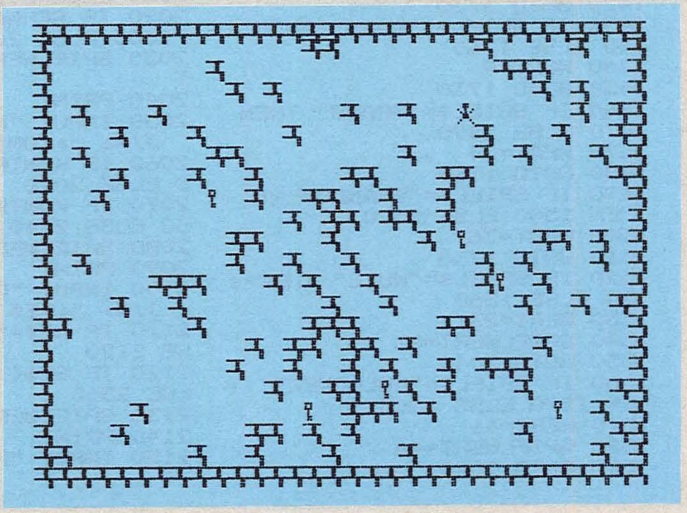
Das flüchtende Männchen wird über die Pfeiltasten (Tasten 5 bis 8) zu den Schlüsseln dirigiert. Die Steuerung ist dadurch erschwert, daß das Männchen immer in Bewegung ist. Von Phase zu Phase erhöhen sich die Geschwindigkeit des Spieles und die Anzahl der Schlüssel.

Eine Besonderheit des Programmes (und wertvolle Hilfe bei ähnlichen Vorhaben) liegt darin, daß die Möglichkeit des Spectrum genutzt wird, den Zeiger auf den Zeichenvorrat in den RAM-Bereich zu verlegen und dort einen eigenen Zeichensatz zu definieren. Die abgedruckte Spielerklärung vermittelt ein Bild der verwendeten Schriftform.

Programmstruktur

0001–0460	Hauptteil des Programmes
1000–1999	Anleitung und Titel
3000–3010	Erweiterung des Schwierigkeitsgrades
9000–9999	Neuer Zeichensatz

André Wieschollek
Tim Harms



Scan: S. Höltgen

Das Computer-Netzgerät

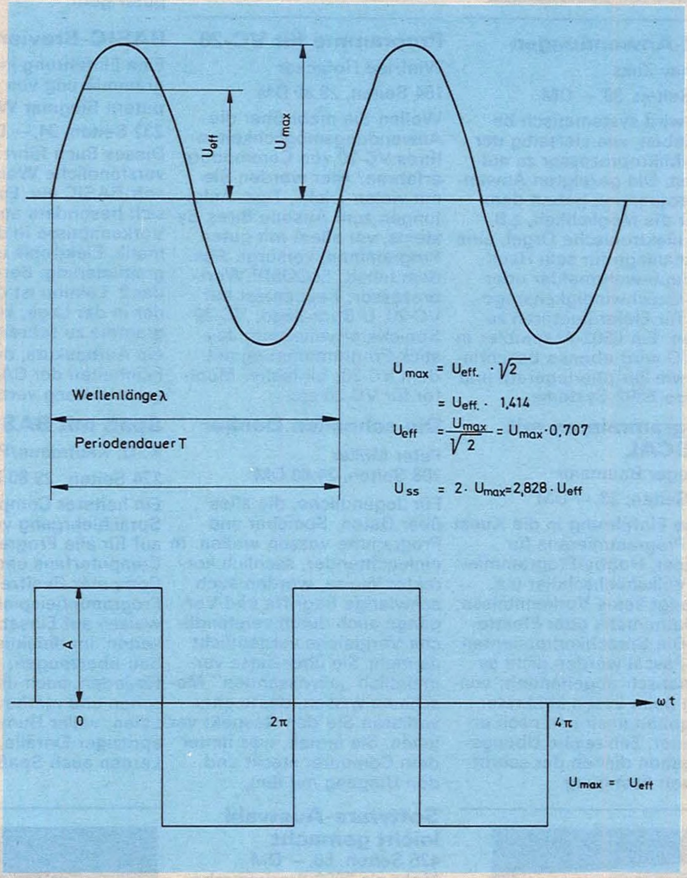
Vorbei ist das Problem der vielen Anschlußkabel mit den verschiedenen Stromversorgungen

Wer eigenhändig in die Hardware greift, braucht sie. Und wer den Home-Computer seines Nachbarn benützen will, findet sie sicher nützlich: die universelle Stromversorgung. Doch die Sache hat auch einen Haken. Die „Erfinder" der Home-Computer haben ihre Geräte so ausgestattet, daß immer nur das Zubehör aus dem jeweiligen eigenen Programm pin-, pegel- und signalkompatibel ist. Aus der Sicht der Computer-Anbieter verständlich, wobei natürlich wirtschaftliche Überlegungen im Vordergrund stehen. Denn jeder Hersteller produziert in erster Linie nach der Devise: „Was andere haben, ist auch bei uns zu finden, nur eben anders und besser. Allerdings jedoch auch ausschließlich auf unser spezielles Programm zugeschnitten." Beim zweiten Grund geht es um die Gewährleistung bei Reparaturen. Einen Rechner zu reparieren, der einen Standard-Fehler aufweist, also einen Fehler, der im Fehlerkatalog jeder gut organisierten Firma gelistet ist, ist im Handumdrehen möglich. Was aber, wenn der Kunde selbst an der Kiste gedreht, gefeilt, geschweißt oder verschlimmbessert hat? Wenn dies schon auf den ersten Blick zu erkennen ist, so können sich die Reparaturkosten noch in Grenzen halten. Ist der „Löter" jedoch schon ein Semiprofessioneller, so ist dem Fehler manchmal gar nicht so leicht beizukommen. Angenommen, das Auswechseln eines integrierten Bausteines kostet 65 Mark inklusive Arbeit und Mehrwertsteuer und Versand. Solche Preise werden pauschal kalkuliert. Allerdings darf hierbei die gesamte Prozedur nicht länger als zehn Minuten dauern: einschalten, überprüfen, aufschrauben, messen, austauschen, überprüfen, zuschrauben, Laufzettel im Firmenrechner ergänzen – fertig. Die Rechnung schreiben zu lassen inklusive Rücksendung erledigt dann der Versand. Diesen kann man noch wesentlich genauer im Zeitablauf kalkulieren. Nehmen wir an, der Fehler erfordert eine Reparaturzeit von 30 Minuten, dann würde das heißen, daß der nächste Rechner in der Zeit „Null" repariert werden müßte. Nachdem diese Möglichkeit nicht praktikabel ist, wird eine Plus-minus-Toleranz in die Normalzeit mit eingerechnet. Wenn die Pluszeit überschritten ist, wird der Fehler nicht separat repariert, sondern die gesamte Rechner-Leiterplatte durch eine Austauschplatine ersetzt. Die nichtreparierte Leiterplatte wandert in die Reparaturabteilung. Sie wird – es liegt auf der Hand – nach erfolgter „Operation" zur Austausch-Leiterplatte. Am Anfang zahlen Sie immer Lehr- und Leergeld. Einmal haben Sie nichts davon, wenn Sie mit der Meßspitze abrutschen und erhalten dann vom Computer-Reparateur eine neue Platine, zum anderen bezahlt indirekt jeder, der sich selbst anlernt. „Ein Transistor kostet 1,40 Mark, gewußt wo, macht 35,40 Mark" – den Spruch kennen wir alle.

Die Auswahl der Spannungen

Das Netzteil können Sie sich so „stricken", wie Sie wollen. Es ist nämlich universell.

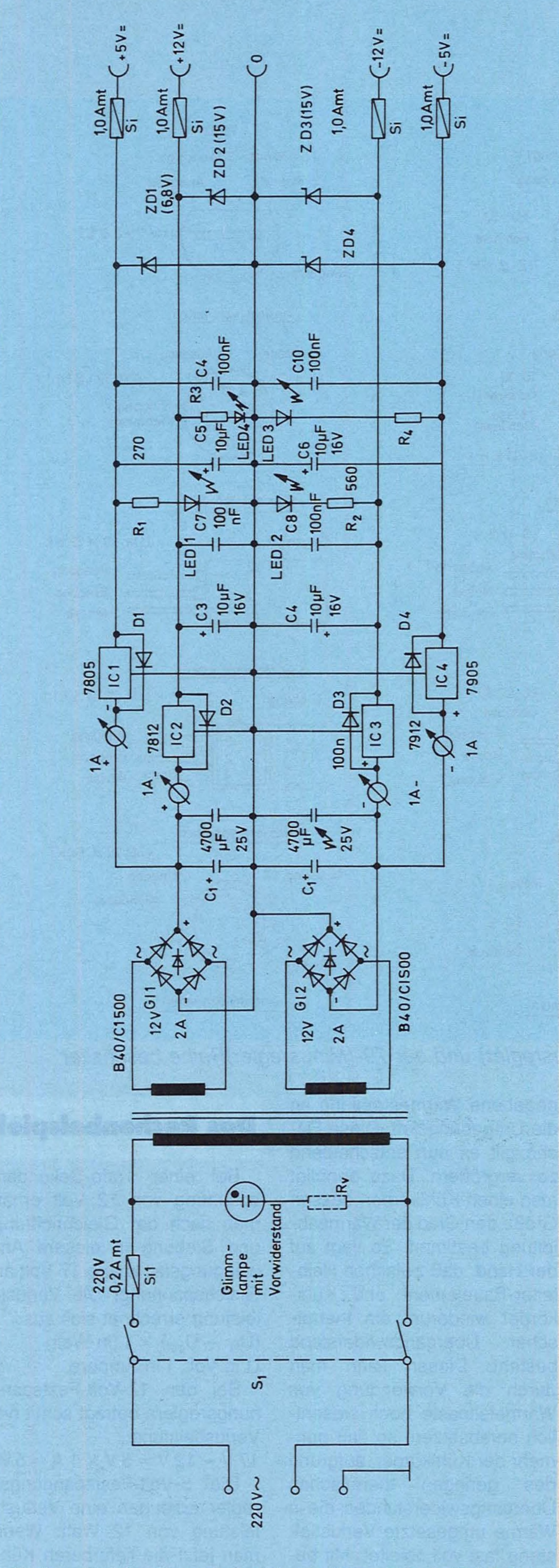
In der Grundausbaustufe liefert es vier Spannungen mit jeweils 1 Ampere Belastbarkeit. Wenn wir uns am TI-Home-Computer orientieren, dann



Sinus- und Rechteckspannung im Vergleich – deutlich sieht man den Unterschied bei der Maximalspannung, denn auf die Zeit kommt es an

benötigt man 12 V, –5 V und +5 V, der Atari möchte gern +5 V haben, der ZX 81 besitzt einen internen +5-Volt-Festspannungsregler und gibt sich mit einer gesiebten Eingangsspannung von 7 bis 11 Volt zufrieden – man könnte da die +5 Volt direkt zuführen und den internen „Heizkörper" abklemmen. Commodore-Rechner werden mit 9 Volt Wechselspannung gespeist. Also ist die interne Versorgungsspannung auch +5 Volt. Bei Dragon ist es ähnlich wie bei den Texas-Instruments-Rechnern, wobei es bei TI zwei Versionen von Netzteilen für den TI 99/4A gibt. Da heißt es aufpassen! Das AC 9900/N ist aufwendig stabilisiert, es liefert also schon die „fertigen" Spannungen zum Rechner, das AC 9900/N2 jedoch kann nur Wechselspannungen abgeben, was bedeutet, daß bei diesem Netzteil die Stabilisierung im Rechner vorgenommen wird. Was ist also mit der vierten Spannung? Die nehme man für den Kassettenrecorder. Im Normalfall reichen da zwischen +6 und 9 Volt aus. Wie man eine Spannung in eine andere umwandelt, wird noch berichtet.

Ein Netzteilaufbau beginnt auf der Primärseite. Metallge-



Professionell präsentiert sich das Gesamtschaltbild des Vierfach-Netzteiles. Die 1A-Instrumente können als Optionen eingebaut werden

häuse müssen über den Schutzkontakt (gelb/grün) geerdet werden. Dann wird ein Zuleitungskabel durch das Gehäuse geführt. Natürlich muß man dazu ein Loch mit ausreichendem Durchmesser in die Rückwand bohren. Innen dient ein Knoten als Zugentlastung. Ein zweipoliger Einschalter trennt im ausgeschalteten Zustand Phase- und Null-Leiter von der Sicherung (Si) und dem Netztransformator.

Der Schalter muß den Strom „können". Es reicht also ein Miniaturschalter für 220 Volt Wechselspannung und für einen Strom von 2 Ampere. Das Produkt von Spannung mal Strom ergibt die Leistung. Wir können also gut und gerne fast ein halbes Kilowatt mit diesem Schalter steuern. Soviel braucht man jedoch in diesem Fall nicht. Die Sicherung dient dazu, den Primärkreis abzusichern. Zur Kontrolle sollte man auf der Primärseite eine Glimmlampe parallel zur Trafowicklung schalten. Dann sieht man gleich, ob die Sicherung entzwei ist und wann Spannung am Trafo liegt. Diese Glimmlampen sind schon für den Frontplatteneinbau mit Vorwiderstand „konfektioniert". Jedes Elektronik-Fachgeschäft hält diese „220-Volt-Glimmlampen" im Sortiment.

Auf der Sekundärseite werden zwei Wicklungen mit 12 Volt/2 Ampere benötigt. Ein bißchen Reserve schadet nie. Mit dieser Dimensionierung erhält man vier Spannungen. Natürlich kann man auch zwei oder mehrere Trafos für dieses Netzteil verwenden. Nur die Spannungen sollten ungefähr den gleichen Spannungswert aufweisen. Wenn der Trafo „mehr Strom kann", so macht das gar nichts. Selbstverständlich können Trafos primär parallel geschaltet werden, sekundär allerdings nur dann, wenn sie wirklich völlig identisch im Aufbau sind. Die Rechner benötigen stabilisierte Gleichspannungen. Also folgt auf der Sekundärseite des Netztransformators zunächst der Gleichrichter. Da die Wechselspannung jedoch mit Sinuskurvenform pulsiert, ergibt sich auch eine pulsierende Gleichspannung. Der Gleichrichter ist in diesem Fall ein Brückgleichrichter. Sein Spannungswert richtet sich nach der höchsten Spitze-Spitze-Spannung $U_{ss} = 2{,}8 \times U_{eff}$.

Ein wenig Theorie

Die Spannungswerte, die man allgemein angibt, sind Effektivspannungswerte. Den Maximal-Spannungswert kann man mit einem hierfür geeichten Voltmeter messen. Ein Oszilloskop beispielsweise zeigt immer den Spitze-Spitze-Wert der Sinusspannung an. Alle sonst handelsüblichen Volt- und Multimeter messen bei sinusförmigen Kurvenzügen den Effektivwert ($0{,}7 \times U_{max}$).

Das liegt an der Kurvenform, denn der Zeitraum, in dem die Spannung „fast immer vorhanden" ist, fällt in den sogenannten Effektivbereich. Würde man rechteckige Wechselspannung messen, ohne es zu wissen, so kann man anstatt 220 Volt – bei unserer Netzspannung – nun die tatsächliche Maximalspannung ($U_{max} = 1{,}4 \times U_{eff}$) anzeigen, da die Rechteckform oben glatt ist und der Effektivwert der Maximalspannung (Spitzenspannung) entspricht.

Der Gleichrichter liefert also eine pulsierende Gleichspannung mit dem Maximalwert von $1{,}4 \times U_{sek}$. Diesen Wert muß der Elektrolytkondensator als Spannungswert mindestens aufweisen. Und dann gilt eine ganz grobe Faustformel: Pro Ampere 2200 µF! An den Elkos mißt man dann die volle Maximalspannung. Auf die Polarität muß man allerdings achten, sonst fliegt einem das Zeug um die Ohren. Und nachher riecht es fatal nach Elko, und die Überbleibsel müssen wieder weggeputzt werden. Außerdem kann der Gleichrichter diese Prozedur unter Umständen übelnehmen und sein Leben abrupt beenden.

Dimensionen

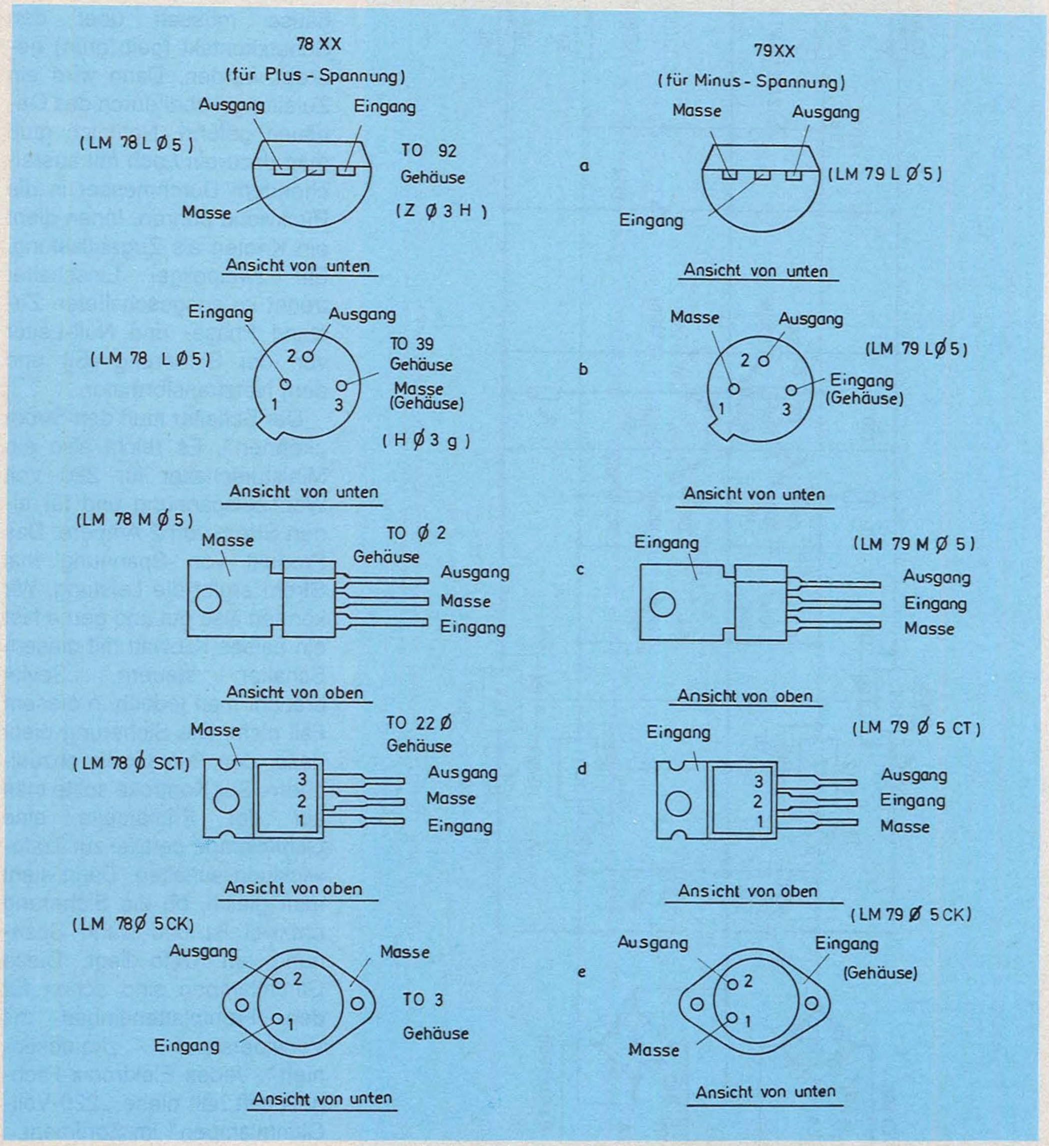
Wer über die Ströme Aufschluß haben möchte, der schalte jetzt ein Amperemeter

Scan: S. Höltgen

in die jeweilige Leitung zwischen Gleichrichter und dem Input-(Eingang-)Anschluß des Festspannungsreglers. Es ist ausreichend, wenn Sie ein Drehspulinstrument mit Güteklasse 2,5 verwenden. Diese Angabe besagt, daß das Instrument beim Vollausschlag einen Fehler von ±2,5 Prozent aufweist. Die Festspannungsregler gibt es in allen Abstufungen für 5, 6, 8, 9, 10, 12, 15, 18 und 24 Volt in Plus(78...)- und Minus(79...)-Ausführungen. Für die 5...18-V-Typen darf die maximal zur Verfügung stehende Eingangsspannung den Wert von 35 Volt nicht übersteigen, bei den 24-Volt-Typen sind 40 V als Maximalspannung noch zulässig. Im allgemeinen reicht es jedoch aus, wenn die Eingangsspannung um 55 Prozent über der gewünschten Ausgangsspannung liegt. Viel höher ist meist nicht sinnvoll, da dann die Verlustleistung ≙ $(U_{In} - U_{Out}) \times I$ (in Watt) zu stark ansteigt. Die über die Festspannungsregler geschalteten Dioden sorgen dafür, daß eine von außen angelegte Gleichspannung und Kondensatorentladungen den Regler nicht zerstört. Die Zenerdioden schützen das „nachfolgende" Gerät vor Überspannung, wenn es passieren sollte, daß der Festspannungsregler „den Geist aufgeben" sollte.

Die Zenerdiode erfüllt jedoch noch eine weitere Funktion. Sie schützt das Netzteil durch Verpolung von außen. Dann wird sie nämlich voll leitend, und die Ausgangssicherung wird zerstört.

Die Kondensatoren am Ausgang der Festspannungsregler unterdrücken dessen schnelles Regelverhalten. Es kann dazu führen, daß der Regler sich mit wilden Regelschwingungen hochschaukelt. Dann oszilliert er.



So werden die ICs der 78-(Plusregler) und der 79-(Minusregler)Reihe beschaltet

Auf die Kühlung kommt es an

Die Differenz von Eingangs- zu Ausgangsspannung – multipliziert mit dem maximal zu entnehmenden Strom – ergibt die Gesamtverlustleistung. Diese Leistung wird im IC in Wärme umgesetzt. Die Verlustwärme muß nun mit einem ausreichend dimensionierten Kühlkörper an die Umgebung abgeleitet werden. Natürlich kann die Wärme der Halbleitersperrschicht nicht voll auf das Gehäuse des Festspannungsreglers übertragen werden. Man spricht hierbei vom thermischen Übergangswiderstand. Bei den verwendeten ICs rechnet man mit 4 C/W (° C/W). Zwischen Gehäuse und Umgebung beträgt er beim T03-Gehäuse 35 C/W, beim T0220-Gehäuse schon 50 C/W. Bedingt ist dieser Unterschied durch die wesentlich kleinere Fläche des zweiten Gehäuses. Man kann sich also merken: Je kleiner die Kühlfläche, desto geringer ist die abgegebene Wärmestrahlung an die Umgebungsluft. Diese Fläche gilt es nun entscheidend zu vergrößern. Dazu benötigt man einen Kühlkörper, dessen Größe den Grad der Wärmeableitung bestimmt. Es liegt auf der Hand, daß zwischen Halbleiter-Bauelement und Kühlkörper wiederum ein thermischer Übergangswiderstand besteht. Diesen kann man durch die Verwendung von Wärmeleitpaste noch wesentlich herabsetzen, so daß nunmehr der Kühlkörper, aufgrund des geringen thermischen Übergangswiderstandes, die in Wärme umgesetzte Verlustleistung fast voll ableitet. Mit berücksichtigt werden muß allerdings der Übergangswiderstand zwischen Sperrschicht und Hableitergehäuse.

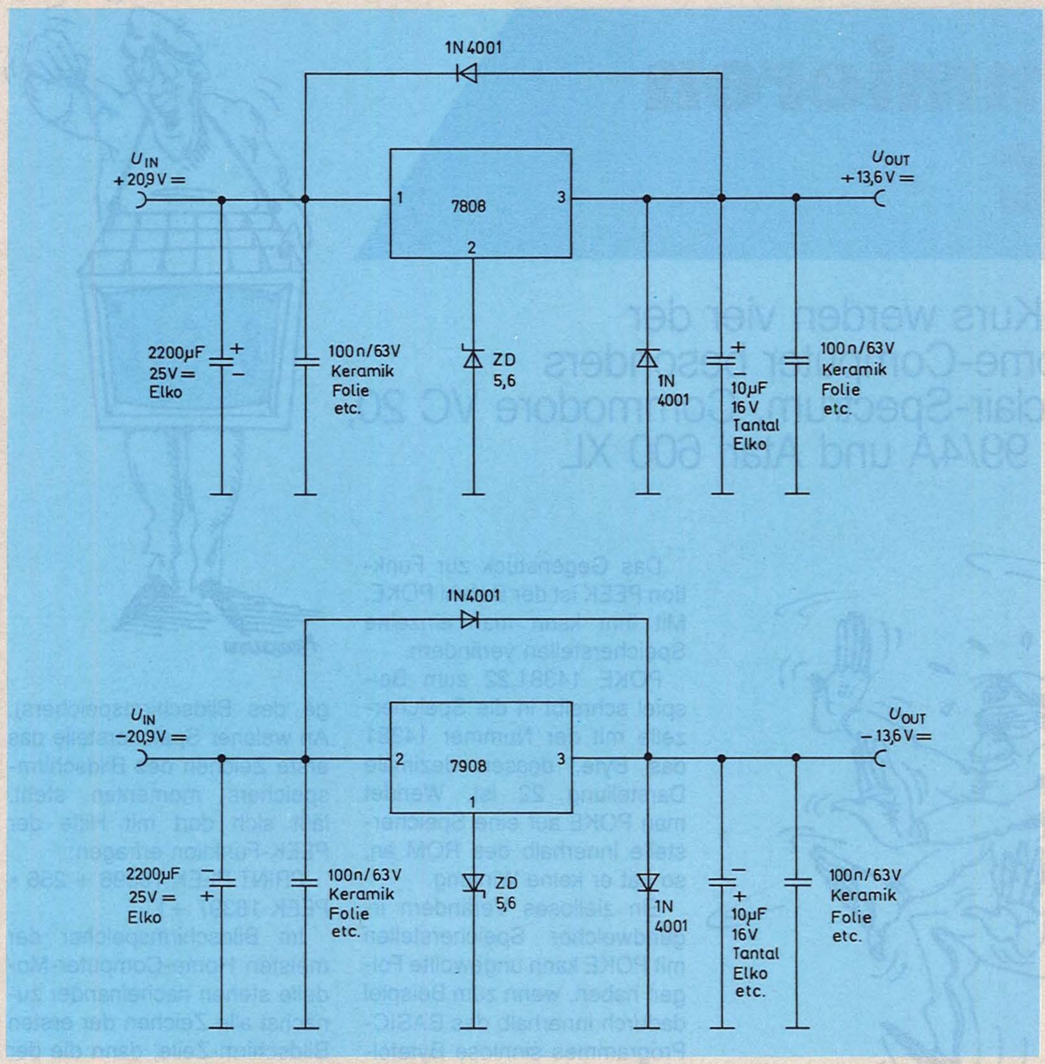
Das Rechenbeispiel

Bei einer Trafo-Sekundärspannung von 12 Volt erhält man nach der Gleichrichtung und Siebung in diesem Anwendungsfall knapp 17 Volt an Gleichspannung. Die Verlustleistung errechnet sich aus:
$(U_{in} - U_{out}) \times I$ (in Watt)
U in Volt, I in Ampere

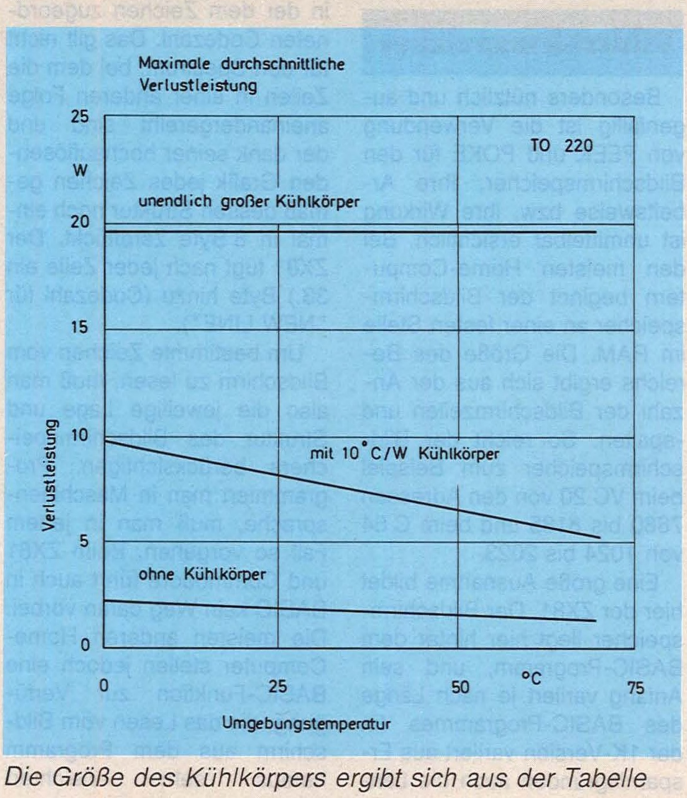
Bei den 12-Volt-Festspannungsreglern beträgt somit die Verlustleistung:
17 V – 12 V = 5 V × 1 A = 5 W

Die 5-Volt-Festspannungsregler erzeugen eine Verlustleistung von 12 Watt. Wenn man jetzt die lieferbaren Kühlerkörper miteinander vergleicht, fällt der Blick auf einen sogenannten Fingerkühlkörper, schwarz eloxiert, für ste-

Scan: S. Höltgen



Die Zenerdiode kann den Regler in seinen Ausgangsspannungsbereich „anheben"



Die Größe des Kühlkörpers ergibt sich aus der Tabelle

hende Montage und mit einem Ableitungsvermögen von 22 C/W. Zählt man in unserem Fall noch die 4 C/W Wärmewiderstand zwischen Halbleitersperrschicht und Festspannungsregler-Gehäuse hinzu und addiert sie mit der maximalen Verlustleistung, so erreicht man einen Wert von 16 C/W.

Bei direkter Montage des Reglers auf den Kühlkörper erhält man einen zusätzlichen Wärmewiderstand von 1,0 bis 1,3 C/W mit dem T0220-Gehäuse. Wird jedoch Sial- oder Silikon-Wärmeleitpaste zwischen Festspannungsregler und Kühlkörper gestrichen – am besten mit dem Messer ganz fein verteilen –, so verringert sich dieser Wert auf 0,6 bis 0,8 C/W. Man erreicht also insgesamt 16,8 C/W und ist daher mit diesem Kühlkörper gut bedient.

Der Aufbau

Sämtliche Bauteile werden laut Stückliste im Handel erworben. Nachdem die Bauteile auf eine große Platine passen, ist es kein Problem, ein Gehäuse zu bauen. Den Bestükkungsplan sowie die fertige Platine können Sie in der HC-Redaktion bestellen. *rf*

Stückliste

Halbleiter
2 B40/C1500 GL 1,2
1 7805 IC1
1 7812 IC2
1 7912 IC3
1 7905 IC4
2 LEDs, rot Plusspannung
2 LEDs, grün Minusspannung
2 ZD 6,8 ZD1,4
2 ZD 15 ZD2,3
4 1N4001 D1...D4

Widerstände
2 270/0,33 W R1,4
2 560/0,33 W R2,3

Kondensatoren
2 4700 µF/25 V Elko C1,2
4 10 µF/16 V Tantal C3...C6
4 100 nF/63 V Keramik C7...C10

Sonstiges
1 Trafo für Printmontage Schaffer KLF 35 VA
1 Glimmlampe für 220 V mit integr. Vorwiderstand
5 Platinen-Sicherungshalter zum Einlöten Si1...S5 Raster: 22,5 mm
4 Drehspulinstrumente 1A (Option)
1 Einschalter S1 (Netz) zweipolig, 250 V/2 A
1 Platine HC-Netzteil HC 601
1 Schraubklemme für Printmontage zweipolig für Netzeingang
4 Fingerkühlkörper 22 C/W
1 Schraubklemme für Printmontage fünfpolig für die Ausgangsspannungen

Rastermaß: Anschluß zu Anschluß beträgt fünf Millimeter

Scan: S. Höltgen

So programmiert man Töne

Jeder kennt das: Man blättert hilflos in der sogenannten Bedienungsanleitung herum, um eine bestimmte Idee zu verwirklichen. Insbesondere bei der Sound-Programmierung sind die Erklärungen meist unzureichend

Die meisten Programme bleiben stumm wie Fische. Das muß nicht sein. Signale können sinnvolle Einrichtungen bei Programmabläufen sein. Klänge und Geräusche erhöhen den Unterhaltungswert von Spielen. Und besonderen Spaß macht es, auf den Tasten des Computers Melodien zu spielen oder Musikstücke einzuprogrammieren, die man selbst wegen ihres Schwierigkeitsgrades gar nicht spielen könnte.

Ebenso ist es möglich, durch die Bewegung des Joysticks Akkorde und Melodienfolgen zu erzeugen, während gleichzeitig auf dem Bildschirm eine Grafik entsteht. Das Wissen darüber, wie jeder auf seinem Computer die Soundmöglichkeiten voll ausnutzen kann, soll hier am Beispiel der drei Rechner TI-99/4A, C 64 und Colour Genie EG 2000 vermittelt werden.

Doch bevor man sich zur Meisterschaft erhebt, muß klar werden, wie man einen Ton aus seinem Computer herausbekommt. Das ist bei jedem System verschieden. Einige Home-Computer erlauben dabei einen relativ einfachen Zugriff auf Sounds, bieten dafür aber wenig Variationsmöglichkeiten in der Klanggestaltung.

Andere bieten einen funktionstüchtigen, mehrstimmigen Synthesizer, sind aber komplizierter zu programmieren. Hinzu kommt, daß in manchen Anleitungsbüchern die Beispielprogramme nicht richtig funktionieren.

Zunächst soll Schritt für Schritt erklärt werden, wie Töne und Akkorde programmiert werden und wie Klänge aufgebaut sind.

Jeder soll durch persönliche Eingaben experimentieren und dabei lernen können.

TI 99/4A

Der TI verfügt über eine stattliche Anzahl von festliegenden Tönen, die über drei Tongeneratoren aufgerufen werden können. Diese Töne sind in ihrer „Klangfarbe" nicht veränderbar. Klangfarben sind die Charakteristiken von Tönen. Zum Beispiel klingt ein „C" auf einem Klavier völlig anders als auf einer Flöte, obwohl es das gleiche „C" ist.

Für den TI bedeutet dies, daß keine Instrumente nachgeahmt werden können. Alle Töne haben die gleiche Klangfarbe, und zwar in der Art eines anhaltenden Orgeltons. Daneben gibt es noch acht verschiedene Rauschtöne, die in ihrer Dauer und Lautstärke beeinflußbar sind.

In der Bedienungsanleitung wird von „weißem" Rauschen und „periodischem" Rauschen gesprochen. Da kein Hinweis darauf gegeben wird, worum es sich dabei handelt, sei dies hier kurz nachgeholt. Weißes Rauschen: Jeder weiß, wenn man die Wellenlängen aller Spektralfarben auf einen Punkt bündelt, entsteht weißes Licht. Ebenso spricht man von weißem Rauschen, wenn alle Frequenzen gleichzeitig erklingen.

Periodisches Rauschen: Hier ertönen zufällig gewählte Frequenzen hintereinander in zeitlich gleichen Teilabschnitten. Periodisches Rauschen klingt nicht so „kratzig" wie weißes Rauschen.

Der erste Ton

Der Zugriff auf die Tongeneratoren ist sehr einfach. Der Befehl lautet CALL SOUND. Danach folgen drei Angaben in Klammern:

1. die Dauer des Tons
2. die Lautstärke
3. die Frequenz (der Ton selbst).

Scan: S. Höltgen

Das Ganze ist dann ein „Sound-Statement".

Für die Dauer eines Tons stehen die Werte 1 bis 4250 zur Verfügung. Außerdem noch die gleichen Werte mit Minus-Vorzeichen. Welche Effekte damit erzielt werden können, wird später behandelt. Ein Ton mit der Dauer von einer Sekunde entspricht dem Wert 1000.

Die Lautstärke ist in den Stufen 0 bis 30 zu regeln, wobei 0 laut und 30 leise ergibt. Bei den Frequenzen können alle Werte von 110 bis 44733 eingegeben werden. Das hört sich viel an. Doch ab der Frequenz von ungefähr 13500 kann das menschliche Ohr keinen Pieps mehr wahrnehmen.

Die Notentabelle in der Bedienungsanleitung auf Seite 85 ist nicht vollständig. Sie können die Liste auf einfache Weise selbst vervollständigen, wenn Sie als Faustregel nehmen: Die Frequenz einer Note verdoppelt sich in der nächsten Oktave. Also hat das „A" mit der Frequenz 1760 in der nächsten Oktave die Frequenz 3520.

Um das mittlere „C" eine Sekunde lang erklingen zu lassen, muß programmiert werden:

```
10 CALL SOUND (1000,262,0)
```

262 ist die Frequenz der Note „C", 0 die größte Lautstärke.

1. Aufgabe: Verändern Sie die Lautstärke von 0 bis 30 nach Belieben und hören Sie sich die Ergebnisse an.

2. Aufgabe: Verändern Sie die Tondauer von 1 bis 4250 nach Belieben und hören Sie sich die Ergebnisse an.

Melodische Spielereien mit Akkorden

Es können beim TI-99/4A drei Töne gleichzeitig erklingen. Genug, um einen Grund-Akkord zu spielen. Das „C" ist bereits bekannt (Frequenz 262); der C-Dur-Akkord besteht aus den Tönen C, E, G bzw. aus den Frequenzen 262, 330, 392. Dieser Akkord kann in ein einziges Sound-Statement eingegeben werden. Dazu müssen Sie nur die Angaben in der Klammer erweitern. Ändern Sie Zeile 10 folgendermaßen:

```
10 CALL SOUND
   (1000,262,0,330,0,392,0)
```

Sie sehen, die Dauer (1000) muß nur einmal angegeben werden; sie gilt für alle drei Töne. Die Lautstärke hingegen muß für jeden Ton einzeln angegeben werden.

Nun könnte es sein, daß Sie einen Sound wünschen, bei dem im ersten Teil nur ein Ton erklingt und danach zwei weitere Töne das Ganze zum Akkord vervollständigen. Das heißt, die Tondauer soll gar nicht für alle drei Töne gleich sein, so wie oben. Diese Aufgabe kann folgendermaßen gelöst werden:

```
10 CALL SOUND (1000,262,0)
20 CALL SOUND
   (1000,262,0,330,0,392,0)
```

Für die Tondauer können natürlich andere Werte genommen werden.

3. Aufgabe: Programmieren Sie weitere Akkorde oder Töne nach Belieben. Zum Beispiel G-Dur (Frequenzen 196, 247, 294), h-moll (Frequenzen 247, 294, 370), A-Dur (Frequenzen 220, 277, 330), d-moll (Frequenzen 294, 349, 440).

Falls Sie die Töne eines Akkords einzeln nacheinander erklingen lassen wollen, programmieren Sie zum Beispiel:

```
10 CALL SOUND (1000,262,0)
20 CALL SOUND (1000,330,0)
30 CALL SOUND (1000,392,0)
```

Der Rechner führt die einzelnen Sound-Statements dann nacheinander aus. Noch hübscher klingt es, wenn die einzelnen Töne nacheinander in den Akkord einklingen:

```
10 CALL SOUND (1000,262,0)
20 CALL SOUND
   (1000,262,0,330,0)
30 CALL SOUND
   (1000,262,0,330,0,392,0)
```

4. Aufgabe: Verändern Sie nach den hier gelernten Prinzipien die Tondauer, die Frequenzen und die Lautstärke der einzelnen Sound-Statements nach Belieben und hören Sie sich die Unterschiede an.

Commodore 64

Die Möglichkeiten der Tonerzeugung auf dem C 64 sind sehr komfortabel – und sehr kompliziert. Dafür kann man praktisch jeden gewünschten Klang oder Geräuscheffekt erzeugen. Sogar die Nachahmung verschiedener Musikinstrumente ist möglich.

Zur Erzeugung eines Tons müssen normalerweise sechs Befehle gegeben werden. Ein weiterer Befehl ist nötig, um den Ton wieder abzustellen. Wenn dieser Abstellbefehl nicht gegeben wird, piept der Computer fortdauernd wie ein hilfloser Wellensittich.

Die Sache mit der Basis-Adresse

Der 64er besitzt drei Tongeneratoren, die einzeln über Basis-Adressen angesprochen werden. Der erste Generator hat zum Beispiel die Adresse 54272. Für die übrigen Elemente eines Tons (Lautstärke, Tondauer usw.) gibt es ebenfalls Adressen.

Alle diese Adressen sind tatsächlich mit Postadressen vergleichbar. Sie sind gewissermaßen wie die Postleitzahlen auf einem Brief. Jeder weiß, daß ein Brief nicht zugestellt werden kann, wenn nur die Postleitzahl angegeben ist. Auch für die Tonerzeugung des C 64 müssen die Adressen vervollständigt werden. „Name", „Straße" und „Hausnummer" sind im übertragenen Sinne Tonhöhe (Frequenz), Klang, Anschlag (hart oder weich) und die Tondauer.

Die wichtigsten Adressen des ersten Tongenerators:

Basis-Adresse des Sound-Chip:	54272 (= BA)
Frequenz Low-Byte	54272 (= BA)
Frequenz High-Byte	54273 (= BA+1)
Wellenform	54276 (= BA+4)
Anschlag + Abschwellen	54277 (= BA+5)
Halten + Ausklang	54278 (= BA+6)
Lautstärke	54296 (= BA+24)

Ab 54279 beginnen die Adressen des zweiten Tongenerators; ab 54286 die des dritten.

Wo das Handbuch irrt

Soll ein Ton erzeugt werden, müssen die Adressen „belegt" werden. Als Beispiel wird der Ton „C" programmiert. In der Tabelle des Handbuchs für den 64er sind auf Seite 158/159 die einzelnen Töne aufgelistet. Das mittlere „C" (Nr. 48) schwingt mit 261,4 Hertz (Hz) pro Sekunde.

Da die Speicherzellen der Adressen nur Zahlenwerte bis zu einer gewissen Größe annehmen, werden alle Schwingungswerte der Töne zerlegt und Low-Byte (Lo) und High-Byte (Hi) genannt. Für das mittlere „C" sind diese Lo/Hi-Werte 103 und 17. Die ersten beiden Programmzeilen lauten:

```
10 POKE 54272,103:REM LO-BYTE
20 POKE 54273,17:REM HI-BYTE
```

Nun zur Lautstärke. Im Handbuch steht, die Lautstärke müsse zuerst gePOKEt werden. Das ist aber in Wirklichkeit nicht nötig. Sie kann auch nach der Eingabe der Lo/Hi-Werte programmiert werden.

Zur Lautstärkenregelung stehen 16 Stufen zur Verfügung, wobei 0 stumm und 15 volle Lautstärke ergibt.

```
30 POKE 54296,15:REM
   LAUTSTAERKE
```

Scan: S. Höltgen

Als nächstes muß der Anschlag eingegeben werden. Ohne Verständnis für den Aufbau eines Tons wird man allerdings mit den Angaben im Handbuch kaum etwas anzufangen wissen.

1 = Anschlag
2 = Abschwellen
3 = Halten (Dauerpegel)
4 = Ausklang

Die komfortablen Programmiermöglichkeiten des C 64 erlauben, alle vier Werte einzeln zu bestimmen. Das bedeutet zwar aufwendige Programmierarbeit, läßt aber viel Spielraum für klangliche Variationen.

Die Adresse für „Anschlag + Abschwellen" (54277) wird mit zwei Angaben belegt. Insgesamt stehen für den Anschlag die Stufen 0∗16 bis 15∗16 zur Verfügung; das Abschwellen stuft sich von 0 bis 15. Im Handbuch ist für das Abschwellen 0∗16 bis 15∗16 angegeben. Glauben Sie das bitte nicht! Es ist ein Druckfehler.

Ein normaler Anschlag ist zum Beispiel 2∗16. Normales Abschwellen zum Beispiel der Wert 5. Die Belegung der Adresse wird folgendermaßen vorgenommen: 2∗16+5. Sie können auch 32+5 nehmen oder gleich alles ausrechnen und 37 angeben. Aber durch diese knappe Angabe verliert man leicht den Überblick, besonders dann, wenn man die Werte noch verändern will. Die nächste Programmzeile in diesem Beispiel lautet:

```
40 POKE 54277,2*16+5 : REM ANSCHLAG
```

Die Belegung der Adresse für „Halten + Ausklang" (54278) erfolgt im Prinzip genauso wie für „Anschlag + Abschwellen". Es stehen die gleichen Abstufungen zur Verfügung, wobei 0∗16+0 stumm + schnell und 15∗16+15 laut + langsam bewirkt.

```
50 POKE 54278,5*16+5 : REM HALTEN
```

Zur Bestimmung der Wellenform (54276) stehen vier Werte zur Verfügung. Auf die Möglichkeiten, bestimmte Instrumente mit Hilfe der Wellenformen nachzuahmen, wird später eingegangen. In folgendem Beispiel wird die „Dreieckswelle" mit dem Wert 17 gewählt:

```
60 POKE 54276,17 : REM WELLE
```

Nun sind die sechs Befehle gegeben, mit denen der Ton „C" auf dem ersten Tongenerator erzeugt werden kann. Starten Sie mit RUN und hören Sie den Ton. Starten Sie noch einmal mit RUN – und Sie werden nichts hören. Das ist eine der typischen Situationen, in denen man ohne Beratung hilflos überlegt, was man denn falsch gemacht hat. Die stumme „Antwort" des Computers kommt daher, weil nach dem ersten RUN der Tongenerator nicht abgeschaltet wurde. Schreiben Sie nun

```
POKE 54276,0
```

und starten Sie mit RETURN. Vervollständigen Sie das Beispiel-Programm, indem Sie schreiben:

```
70 FOR X=0 TO 1000 : NEXT X
80 POKE 54276,0 : REM ABSCHALTEN
```

Zeile 70 sorgt für eine Verzögerung. Ohne sie würde gar nichts hörbar sein, da der Computer bei seiner schnellen Arbeitsweise im gleichen Moment nach dem Anschalten des Tongenerators ihn durch Zeile 80 wieder ausschalten würde.

Das richtige Gefühl für den guten Ton

1. Aufgabe: Setzen Sie in Zeile 60 statt 17 die Werte 33 und 129 ein und hören Sie sich die Ergebnisse an. 33 erzeugt die sogenannte Sägezahnwelle, 129 sorgt für ein Rauschen.

2. Aufgabe: Ändern Sie in Zeile 40 die Aufgaben hinter der Adresse in beliebiger Weise, wobei Sie zwischen 0∗16+0 bis 15 und 15∗16+0 bis 15 wählen können. Ab etwa 10∗16 werden Sie den vollen Klang nur noch hören können, wenn Sie in Zeile 70 die Verzögerung von 1000 auf 10000 erhöhen.

3. Aufgabe: Ändern Sie in Zeile 50 die Angaben hinter der Adresse in beliebiger Weise, wobei Ihnen die gleichen Wahlmöglichkeiten wie bei der zweiten Aufgabe beschrieben, zur Verfügung stehen.

Durch diese Änderungen bekommen Sie ein Gefühl dafür, in welch vielseitiger Weise ein einziger Ton verändert werden kann, denn es handelt sich immer noch um das „C".

Colour Genie EG 2000

Von den hier vorgestellten Home-Computern besitzt der Colour-Genie EG 2000 besonders komfortable Sound-Programmiermöglichkeiten, da dieses System zum einen den einfachen Zugriff auf festliegende Töne erlaubt, darüber hinaus aber auch breite Manipulationsmöglichkeiten von Tönen und Geräuschen zuläßt, die allerdings etwas komplizierter zu programmieren sind.

Interessante Effekte lassen sich außerdem mit acht verschiedenen Klangwellenformen erzielen. Aber noch bevor man sich über diese Klangvielfalt freuen kann, kommt eine Enttäuschung. Denn der Benutzer ist mit diesen Möglichkeiten ziemlich allein gelassen. Besonders für die kompliziertere Programmierung des sogenannten PSG (Programmable Sound Generator) fehlt es quasi an jeglichen Bedienungs-Hinweisen.

Im Manual wird falsch gespielt

Der Zugriff auf die festliegenden Töne erfolgt durch den Befehl PLAY und vier Angaben in Klammer: Kanal, Oktave, Note und Lautstärke. Unter Kanal ist jeweils einer der drei Tongeneratoren zu verstehen. Die Angabe kann also 1, 2 oder 3 lauten. Für die zweite Angabe stehen acht Oktaven zur Verfügung (1–8). Jede Oktave ist in zwölf Einzeltöne unterteilt. In der Bedienungsanleitung steht, daß dabei die Werte 1 bis 7 der E-Dur-Tonleiter entsprechen. Das ist Unsinn. Die E-Dur-Tonleiter enthält vier „Erhöhungen" (Cis, Fis, Gis und Dis). Die Werte 1 bis 7 entsprechen vielmehr der C-Dur-Tonleiter:

1 = C | 2 = D | 3 = E
4 = F | 5 = G | 6 = A
7 = H

Die Werte 8 bis 12 entsprechen den „Erhöhungen":

8 = Cis | 9 = Dis | 10 = Fis
11 = Gis | 12 = B

Die Lautstärke ist von 0 bis 15 abstufbar, wobei 15 den lautesten Klang ergibt. Nach diesen Informationen kann das mittlere „C" programmiert werden:

```
10 PLAY (1,3,1,15) : REM "C"
```

Da nun nach RUN der Ton nicht wieder verklingt, ist es sinnvoll, zuvor noch eine Verzögerungs-Schleife einzubauen und den Befehl zum Abstellen des Tongenerators zu geben (vergleiche Commodore 64, Programmzeile 70).

```
40 FOR X=0 TO 1000:NEXT X
50 PLAY (1,3,1,0) : REM ABSCHALTEN
```

Die Zeilennummern 40 und 50 wurden gewählt, weil nachfolgend noch weitere Sounds in die Zeilen 20 und 30 zwischengeschoben werden.

Akkorde nach Belieben

Soll der Akkord „C-Dur" erklingen, der aus C,E,G besteht, muß zusätzlich eingegeben werden:

```
20 PLAY (2,3,3,15) : REM "E"
30 PLAY (3,3,5,15) : REM "G"
```

und zum Ausschalten zusätzlich:

```
60 PLAY (2,3,3,0)
70 PLAY (3,3,5,0)
```

Scan: S. Höltgen

1. Aufgabe: Spielen Sie andere Akkorde nach Belieben, zum Beispiel D-Dur (D, Fis, A), d-moll (d, f, a), F-Dur (F, A, C). Dur-Tonarten werden immer in Großbuchstaben geschrieben; Moll-Tonarten in Kleinbuchstaben.

Die sieben Siegel des PSG

Wie bereits gesagt, sind im Anleitungsbuch des Colour-Genie die Möglichkeiten des Programmable Sound Generators so gut wie gar nicht erläutert. Dabei wäre das besonders wichtig gewesen, da von den vielen ansprechbaren Sound-Registern nur bestimmte Kombinationen von Befehlen korrespondieren und andere ohne jede Wirkung bleiben.

Für die drei Tongeneratoren steht neben einer Grobabstimmung (Werte 0 bis 15) eine Feinabstimmung der Frequenzen zur Verfügung (Werte 0 bis 255). Da nirgendwo steht, welche Frequenzen hinter diesen Werten verborgen sind, hier wenigstens einige Töne:

255 = Note A, 4. Oktave
127 = Note A, 5. Oktave
63 = Note A, 6. Oktave
31 = Note A, 7. Oktave
15 = Note A, 8. Oktave

Register 6 organisiert die Rauschfrequenz, wobei 0 einen reinen Ton und 31 das gröbste Rauschen ergibt. Die Rauschfrequenz muß in Register 7 aktiviert werden. Die komplizierte Bit-Belegung des Registers 7 wird später erklärt.

Register 8 hat doppelte Funktion. Zum einen stuft es die Lautstärke ab (Werte 0 bis 15), wobei 0 stumm und 15 laut ergibt; zum anderen öffnet es mit dem Wert 16 den Zugang zur Modulation der Hüllkurve, also die Gesamtgestalt eines Klanges. Die Werte aller vier Elemente des Tons ergeben die Hüllkurve.

Register 11 erlaubt die Festlegung der gewünschten Frequenz in der Abstufung 0 bis 255, also genau wie in Register 0. Wichtig: Wenn mit Hüllkurve gearbeitet wird (Register 8, Wert 16), muß der gewünschte Ton in Register 11 gewählt werden. Wird ohne Hüllkurve gearbeitet (Register 8, Wert 0 bis 15), muß der gewünschte Ton in Register 0 gewählt werden. Diese Angaben beziehen sich auf den Tonkanal A, gelten aber auch für die anderen Kanäle mit den entsprechenden Register-Nummern. Nehmen Sie zur Veranschaulichung folgendes Beispiel:

```
10 SOUND 0,127 : REM A, 4. OKTAVE
20 SOUND 6,0 : REM KEIN RAUSCHEN
30 SOUND 7,8 : REM KEIN RAUSCHEN
40 SOUND 8,15 : REM LAUT
```

Auch hier eine Verzögerungs-Schleife und Sound-Abschaltung (Vergleiche Commodore 64, Zeile 70).

```
100 FOR X=0 TO 1000 : NEXT X
110 SOUND 8,0 : REM ABSCHALTEN
```

Nach RUN hören Sie den gewählten Ton. Ändern Sie:

```
40 SOUND 8,16
```

Nach RUN hören Sie nichts. Sie haben die Hüllkurve gewählt, ohne Frequenzangabe (Register 11) und ohne Hüllkurvenform (Register 13). Erweitern Sie das Programm:

```
50 SOUND 11,127 : REM HÜLLKURVE
60 SOUND 13,13 : REM WELLENFORM
```

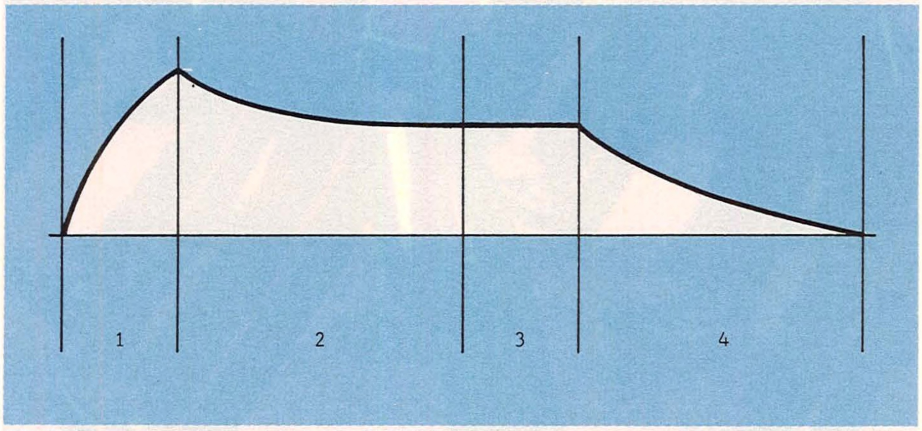
Für die Wellenform der Hüllkurve stehen acht Werte zur Verfügung: 0,4,8,10,11,12,13,14. Diese Wellenformen werden in Register 13 aufgerufen.

Experimente mit Wellenformen

2. Aufgabe: Probieren Sie alle Wellenformen durch und hören Sie sich die Ergebnisse an.

3. Aufgabe: Arbeiten Sie mit Rauscheffekten, indem Sie bei Zeile 20 die Werte ändern (6, 1 bis 31). Zur Aktivierung des Rauschens muß Register 7 (Zeile 30) geändert werden, und zwar in 7,0 oder 7,64.

Alfred Görgens



Die Zerlegung eines Tones in vier Phasen

	TI-99/4A	C 64	Colour Genie EG 2000	Atari 800 Atari 600 XL
Anzahl der Tongeneratoren	3	3	3	4
Tonumfang	8 Okt.	8 Okt.	8 Okt.	3½ Okt. (bei BASIC) 5½ Okt. (bei System)
Aufruf der Tongeneratoren	CALL SOUND	POKE	a) PLAY b) SOUND	a) SOUND b) POKE
Handhabung	einfach	kompliziert	a) einfach b) kompliziert	a) einfach b) kompliziert
Melodien programmieren	einfach	kompliziert	a) einfach b) kompliziert	a) einfach b) kompliziert
Geräusche und Effekte	möglich, aber unbefriedigend	komfortabel	komfortabel	möglich, aber kompliziert
Veränderung der Klangfarbe	–	komfortabel	unbefriedigend	möglich, aber kompliziert

Die Sound-Möglichkeiten von vier Home-Computern im Überblick

Scan: S. Höltgen

Der Byte-Sprinter

Wer ist der schnellste – acht Benchmark-Tests überprüfen sechs Computer auf ihren internen Datenfluß

Die erste Pause hat gerade begonnen. Auf dem Schulhof stehen die Schüler der Klasse 8a nebeneinander. Der Klassensprecher, groß und schlaksig, baut sich vor den anderen auf, malt eine Marke an die Wand der Schule und meint: „Keiner von euch ist größer als ich". Die anderen sind verdutzt. Sie stellen sich der Reihe nach auf und vergleichen ihre Größen. Tatsächlich, Manfred ist der größte. Manfred hat, schlicht und ergreifend ausgedrückt, eine Benchmark an die Wand gemalt. Allerdings, und das sei ganz ausdrücklich betont, dieser Maximalwert, mit dem andere Größen verglichen werden, sagt nichts darüber aus, wer von den Schülern der Klasse 8a der schnellste Läufer oder der beste Schwimmer ist.

Für Computer gelten diese geschilderten Wettbewerbsbedingungen analog. Die bekanntesten Benchmark-Tests im Klein-Computer-Bereich sind nun einmal die Geschwindigkeitsvergleiche. Sicher sind auch andere Spezies überprüfbar, etwa die Rechengenauigkeit oder die für ein „Normprogramm" notwendige Größe des Speicherbereiches.

Kilobaud-Programme

Als es die ersten Klein-Computer am Markt gab, war natürlich das Interesse groß, wer der schnellste von ihnen sei. Deshalb rief die damals sehr bekannte amerikanische Zeitschrift „Kilobaud-Microcomputing" die Benchmark-Tests ins Leben. Acht verschiedene Testprogramme mußten die „Byte-Denksportler" absolvieren, bevor eine Entscheidung über ihre Datenspurtfähigkeit getroffen werden konnte. Diese Tests sind nun für Home-Computer umgeschrieben worden und von BM1 bis BM8 numeriert. In der Tat ermöglichen sie ein Austesten des BASIC-Interpreters auf seine Geschwindigkeit bei der Programmausführung. Die Benchmark-Software ist Schritt für Schritt (BM1 bis BM8) aufeinander aufgebaut. Im Klartext bedeutet dies, daß nach der Eingabe des ersten Programmes das folgende durch einfaches Einfügen weiterer Programmschritte erzielt wird. So entsteht BM3 aus BM2 durch Einfügen der Programmzeile 510 – so einfach ist das.

Scan: S. Höltgen

Auf die Plätze, fertig ...

Bevor es richtig losgeht, müssen Sie Ihre Stoppuhr auf Null setzen. Das Starten der Benchmark-Tests wird am Bildschirm durch ein „S" für Start angezeigt. Jetzt heißt es aufpassen nach der Devise: Finger am Stoppdrücker, Auge am Bildschirm. Da ist es nämlich schon, das „E" wie Ende gut, alles gut. Wer allerdings seine Reaktionszeit bezweifelt, der kann die K-Endwerte beim BM1-Programm in Zeile 400 von 1000 auf 10 000 heraufsetzen. Damit erhält man dann statt 1,4 Sekunden den zehnfachen Betrag, ist allerdings dann auch um diesen genauer. Wenn die Zehntelsekunde gewünscht wird, dann können Sie sich bis 100 000 hochschrauben.

Wer Programme genau miteinander vergleicht, wird sofort wissen, warum der achte Test so kurz verläuft. Das liegt daran, daß die Schleife, die sonst einen 1000maligen Durchlauf absolvieren muß, hierbei nur 100mal abgearbeitet wird.

Ausnahmen bestätigen die Regel. So versteht eben nicht jeder Home-Computer das gleiche BASIC. Bei den Benchmark-Tests 5, 6 und 7 ist bei Sinclair-Rechnern END durch STOP zu ersetzen. In der Zeile 530 des achten Programms ist mit dem Zeichen ↑ die Potenzierung gemeint. Außerdem gilt es, zwischen THEN und 500 in der Zeile 600 beim Sinclair-ZX 81 und Spectrum und Atari 600 XL ein GOTO einzufügen. *– rf*

Benchmark-Test

	1	2	3	4	5	6	7	8	
ZX 81	17,7	27,2	54,1	63,2	74,2	199,9	276,2	92,0	
Spectrum	4,7	8,5	16,8	19,5	23,1	52,7	77,1	24,2	
VC 20	1,4	8,2	15,3	16,9	18,2	26,9	42,4	9,9	Sek.
C 64	1,5	9,5	18,0	19,9	21,5	31,9	50,5	11,7	
TI-99/4A	2,9	8,9	24,0	24,6	26,2	61,9	84,8	41,1	
Atari 600 XL	2,3	7,3	16,1	22,9	26,4	40,1	60,7	44,9	

Die ermittelten Zeiten wurden „von Hand" gemessen. Dabei bedeutet die Angabe 1.03,2 eine Minute, drei Sekunden und zwei Zehntelsekunden

```
10 REM BM1
300 PRINT "S"
400 FOR K=1 TO 1000
500 NEXT K
700  PRINT "E"

10 REM BM2
300 PRINT "S"
400 K=0
500 K=K+1
600 IF K<1000 THEN 500
700 PRINT "E"

10 REM BM3
300 PRINT "S"
400 K=0
500 K=K+1
510 A=K/K*K+K-K
600 IF K<K1000 THEN 500
700 PRINT "E"

10 REM BM4
300 PRINT "S"
400 K=0
500 K=K+1
510 A=K/2*3+4-5
600 IF K<1000 THEN 500
700 PRINT "E"

10 REM BM5
300 PRINT "S"
400 K=0
500 K=K+1
510 A=K/2*3+4-5
520 GOSUB 820
600 IF K<1000 THEN 500
700 PRINT "E"
800 END
820 RETURN

10 REM BM6
300 PRINT "S"
400 K=0
430 DIM M(5)
500 K=K+1
510 A=K/2*3+4-5
520 GOSUB 820
530 FOR L=1 TO 5
540 NEXT L
600 IF K<1000 THEN 500
700 PRINT "E"
800 END
820 RETURN

10 REM BM7
300 PRINT "S"
400 K=0
430 DIM M(5)
500 K=K+1
510 A=K/2*3+4-5
520 GOSUB 820
530 FOR L=1 TO 5
535 M(L)=A
540 NEXT L
600 IF K<1000 THEN 500
700 PRINT "E"
800 END
820 RETURN

10 REM BM8
300 PRINT "S"
400 K=0
500 K=K+1
530 A=K↑2
540 B=LOG(K)
550 C=SIN(K)
600 IF K<100 THEN 500
700 PRINT "E"
```

Benchmark-Tests sind immer gefragt. Hier die erforderlichen Programme

Scan: S. Höltgen

Horror-Trip im Geisterschloß

Revolution auf dem Bildschirm: Eine neue Welle von Computerspielen stellt alles in den Schatten, was bisher elektronisch die Zeit vertrieb – die „Adventure Games" kommen

Scan: S. Höltgen

Die Zeit der Killer ist vorbei, die Scharfschützen können endlich in Pension gehen und ihre Schießprügel an die Wand hängen. Jetzt zählen im Kampf ums Überleben nicht mehr der schnelle Finger am Abzug oder der starre Blick aufs Punktekonto, sondern nur noch Intelligenz, Gedächtnis, Phantasie und Kombinationsfähigkeit: Eine neue Sippe von Computerspielen macht Schluß mit der öden Ballerei auf außerirdische Angreifer oder feindliche Panzer. Aber auch PacMan, King Kong & Co. dürften in Zukunft einen schweren Stand haben – die „Adventure Games" kommen, und damit das aufregendste und nervzerfetzendste Vergnügen, das die kühlen Rechner dem Spiele-Freak jemals zu bieten hatten.

Kein starrer Programmablauf langweilt nach kurzer Zeit, der Ablauf des Geschehens auf dem Bildschirm wird direkt vom Handeln des Herausforderers beeinflußt. Zur Reise in die geheimnisvolle Welt der „Adventure Games" benötigt man viel Geduld, einen Atari-Home-Computer mit mindestens 32-KB-RAM sowie eine Disketten-Station. Die vorgestellten Spiele „Galahad" und „Schloß des Grauens" kosten je 49 Mark – der Preis für ein schier endloses Vergnügen.

„Galahad" versetzt den Spieler in eine mittelalterliche Landschaft. Sir Galahad, der heldenhafte Ritter aus der Tafelrunde von König Arthur, erscheint als kleine Spindel, die mit dem Joystick gesteuert wird. Mit dem Feuerknopf lassen sich verschiedene Gegenstände „aufheben".

Gefahr und Belohnung

Das Land, in dem Sir Galahad den „Heiligen Gral" suchen muß, ist durch verschiedene Farben in Wälder, Wiesen, Flüsse und Meere aufgeteilt. Es gibt schwarze und weiße Burgen, in denen Gefahr und Belohnung warten. Undurchdringliche Wände begrenzen das Spielfeld. Im Wald bewegt sich der Spieler langsamer als auf den eingezeichneten Wegen. Findet der kühne Ritter ein Pferd, kommt er noch schneller voran. Doch Eile ist nicht geboten. Kluge Umsicht führt bei diesem Abenteuer viel eher zum Ziel.

Flüsse durchziehen das Spielfeld. Schmale Bäche können gefahrlos durchquert werden. Über breite Flüsse führen Brücken den Weg. Im übrigen ist die Berührung mit Flüssen oder dem Meer tödlich. Der Ritter zerplatzt mit einem Knall und findet sich am Ausgangspunkt wieder.

Eine Tarnkappe schützt den Helden vor mancherlei Monstern, die der Computer per Zufall ins Feld schickt. Ein geheimnisvoller Schlüssel sollte nicht unbeachtet bleiben.

Das Geheimis der Burgen

Die Grafik ist recht großzügig angelegt und in Teile angeordnet. Gelangt der Spieler an den Bildschirmrand, springt die Grafik in das nächste Planquadrat über. 16 Bildschirme bilden das gesamte Spielfeld. Vier Burgen bergen ihr Geheimnis, das sie nur nach vielen vergeblichen Versuchen, nach langem Rätseln und verzweifeltem Grübeln preisgeben. Manche Falle wartet auf den Ahnungslosen, um ihn wieder zum Start zurückzuwerfen. Nur eine Burg öffnet endlich einen geheimen Gang, der auf eine neue Spielebene führt. Und wieder steht der Held am Anfang seiner Suche, muß Bildschirm um Bildschirm des neuen Landes durchforschen.

Das Rätsel zu lösen und den Heiligen Gral zu finden, kostet Stunden voller Versuche und Kopfzerbrechen. Endlos oft muß der tapfere Ritter vom Start beginnen, weil ihn die verborgenen Gefahren übermannt haben. Nur systematisches Vorgehen ist sinnvoll. Vom erforschten Gelände sollte man sich eine Karte zeichnen und über gefundene Gegenstände, Fallen und Gefahren Buch führen. Und wer sich nicht entmutigen läßt, wird irgendwann sehen dürfen, wie sich der Programmierer Doug Crockford den Heiligen Gral vorstellt.

Abgesehen von den Spielgeräuschen, die das Geschehen untermalen, laufen Echtzeit-Abenteuer sprachlos ab. Das Rätsel wird durch die Bewegung des Spielers und verschiedener Gegenstände gelöst. Bei einer anderen großen Gruppe von Adventure-Spielen ist das ganz anders.

Die Dialog-Abenteuer bestehen nur aus Text, der auf dem Bildschirm erscheint und vom Spieler wie ein Buch gelesen wird. Allerdings kann sich der Spieler in den „Kapiteln" frei bewegen, er kann, ja, er muß handeln, kann Dinge verändern und so langsam dem Rätsel auf die Spur kommen. Manche Dialog-Abenteuer zeigen zusätzlich zum Text eine Grafik. Das ist als Gedächtnisstütze ganz nützlich, für das eigentliche Spiel aber nicht von Bedeutung. Übereinstimmendes Merkmal dieser Spielform ist, daß der Spieler sich beliebig viel Zeit nehmen kann, genau wie man ein Buch jederzeit beiseite legen kann.

Dialog-Abenteuer sind im angloamerikanischen Raum schon eine ganze Weile der große Renner. Hier bei uns konnten sich nur wenige so richtig dafür erwärmen. Dafür gibt es zwei Gründe. Einmal ist der Computer nicht besonders redegewandt, oder vielmehr: er hört nicht gut. Er versteht nur die wenigen Worte, die ihm der Programmierer eingegeben hat. Tippt der verzweifelte Abenteurer ein Wort ein, das nicht im Daten-File des Programms zu finden ist, reagiert der Computer mit Floskeln wie „Ich verstehe Sie nicht" oder ähnlichem. Will man etwas Bestimmtes erreichen, kann es recht entnervend werden, bis man endlich nach langem Suchen das richtige Wort gefunden hat.

Abenteuer im Weltraum

Da Computer-Programme fast ausschließlich importiert werden, kommt für den deutschen Spieler noch die Hürde der fremden Sprache dazu. Da hört für viele dann eben der Spaß auf.

„Abenteuer im Weltraum" (Atari) spricht unsere Sprache. Der Spieler findet sich in der Kommandozentrale eines Raumschiffs wieder. Er ist der einzige Überlebende, und sein Auftrag lautet, das Ei des Alien zu finden und ins Raumschiff-Laboratorium zurückzubringen, um ein Unheil zu verhindern. Durch die Kommandos N, W, O und S für die Himmelsrichtungen und H für hoch und R für runter bewegt sich der Spieler.

Der Dialog mit dem Computer läuft in der Form Objekt – Verb ab, etwa „Kassette abschließen" oder „Lampe nehmen".

Ein kurzer Text beschreibt den Raum. Dabei kann jedes Wort ein Hinweis sein. Durch die Eingabe einer Bewegungsrichtung gelangt der Spieler in den nächsten Raum, der wieder kurz beschrieben wird. Wer seine Erkundungen nicht in einem Lageplan festhält, irrt nur sinnlos umher.

In der Kommandozentrale des Raumschiffs gibt es eine Luke, die läßt sich aber nicht öffnen. Auch der Zugang zum Reaktorraum birgt sein Geheimnis. Die Mütze des Kapitäns liegt im wasserlosen Duschraum. Im Billardzimmer liegt ein Buch. Enthält es einen Hinweis? Im Computer-Zentrum liegt ein strahlenfester Handschuh, doch der Schacht, der dorthin führt, muß erst geöffnet werden. Im Zimmer des Kommandanten verbirgt ein Geheimfach eine Nachricht. Und irgendwie gehören alle diese Steinchen zu einem großen Mosaik.

Scan: S. Höltgen

Alles andere als gemütlich: Das „Schloß des Grauens" steckt voller Geheimnisse – und der Besucher in einer (fast) aussichtslosen Situation

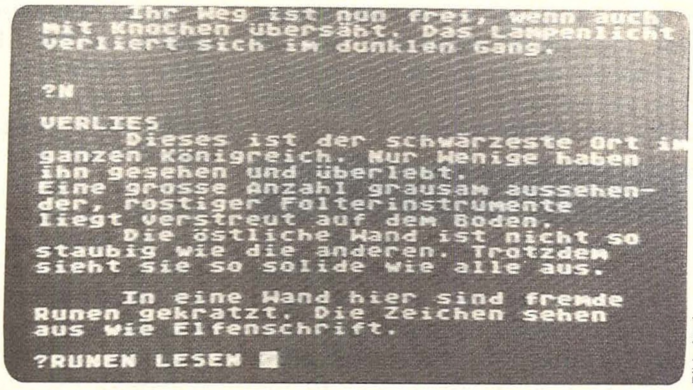
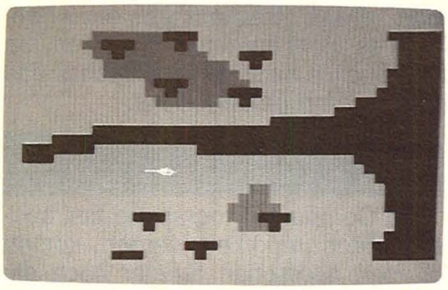


Bild: Atari



Wo ist der Heilige Gral? Sir Galahad (in Weiß) reitet durch die Lande

Wer nach schlaflosen Nächten des Rätsels Lösung herausgeknobelt hat und endlich die Luftschleuse des Raumschiffs verlassen kann, muß auf dem fremden Planeten noch manche Gefahr bestehen. Ein unbedachter Schritt ins Ungewisse, und der eifrige Forscher stürzt in eine bodenlose Grube. Das Programm verabschiedet sich mit dem hämischen Satz, daß nur der SYSTEM-RESET-Knopf ein Ausweg wäre, aber das bedeutet Neubeginn des ganzen Spiels.

Das Schloß des Grauens – eine harte Nuß

Hat man endlich in der Stalaktitenhöhle mit dem richtigen Gegenstand den Alien in die Flucht geschlagen, schnappt man sich das Ei, eilt zurück zum Raumschiff und stellt fest, daß die Luftschleuse unerreichbar hoch ist. Sollte man da vielleicht im Geräteraum einen wichtigen Gegenstand falsch eingeschätzt haben?

Unsere Großmutter hat gemahnt, daß wir uns nie mit einem bösen Zauberer anlegen sollen. Aber wer nicht hören will, muß spielen. Und „Schloß des Grauens" (Atari) ist wirklich eine grauenhaft harte Nuß!

Der Spieler findet sich in einem Schloßhof wieder, in dem eine Fontäne friedlich plätschert. Die Aufgabe besteht nur darin, das Schloß wieder zu verlassen. Allerdings ist zu bedenken, daß es sich um das Häuschen eines verrückten Zauberers handelt. Doch das merkt man beim Herumirren schnell. Der Spielverlauf ist wie gehabt. Ein kurzer Text beschreibt den Raum. Der Computer reagiert auf die üblichen Kurzsätze. Den nächsten Raum erreicht man durch Eingabe der Bewegungsrichtung N, W, S, O, R oder H (Runter oder Hoch).

Um den Schloßhof sind acht Räume gruppiert. Eine große Treppe führt in die erste Etage, wo noch einmal acht Räume zu durchstöbern sind. Dann kommen natürlich noch diverse Turmkammern und Kellergewölbe hinzu.

Drei Wege in die Freiheit bieten sich an: Das große Burgtor ist verschlossen und die Kurbel läßt sich nicht drehen; ein kleines Gitter ist mit einer Kette verschlossen; im ersten Stock ist ein Fenster mit einem schweren Eisengitter versperrt. Wer nicht viel Grips im Kopf hat, wird nur verstört durch die Zimmer irren und die Welt nicht mehr verstehen.

Im Salon flüstert eine Pflanze nach Wasser. Ein Lunch-Paket ist in ein Pergament eingewickelt. Jemand soll ein großer Tierliebhaber gewesen sein, und im Glockenturm findet sich ein Beutel mit Vogelfutter. Die Anweisung „Vogelfutter essen" ist nicht zu empfehlen, es schmeckt nicht besonders. Da ist dieser kleine Kuchen im Zimmer der Königin schon interessanter. Sein Verzehr vermittelt ein „elfisches" Gefühl. Ist das gefährlich? Und wie steht es mit der Flasche Wasser? Vielleicht die flüsternde Blume ..

Ein Gerippe im Kerker

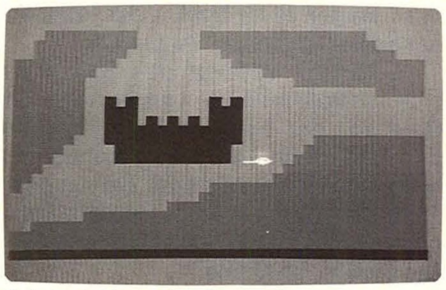
Auch das mysteriöse Lunch-Paket läßt sich verspeisen. Ein Computer kann sich schließlich nicht den Magen verderben. Die Spur zur Lösung des Rätsels beginnt im Zimmer der Mätresse. Der alte König war ein Lebemann, doch die Indizien seiner Seitensprünge hat er in einem geheimen Zimmer versteckt. Ein Amulett ist der Schlüssel dazu. Doch wo ist es zu finden und was ist damit anzustellen?

Im stockfinsteren Schloßkerker versperrt ein Gerippe den Weg und will ihn einfach nicht freigeben. Mit keinem Wort ist es zu überreden. Hat man sich endlich doch den Weg geebnet, steht man vor einem neuen Rätsel: eine Inschrift, die nichtssagend bleibt, weil sie in Runen verfaßt ist. Natürlich gibt es einen Trick, und die Bedeutung wird enthüllt, nur ist man dann genauso schlau wie vorher.

Ein kleiner Tip von einem leidgeprüften Abenteurer: In dem geheimnisvollen Satz fehlt genau genommen ein Komma!

„Sie sind ein Abenteurer!"

Drei geheime Wörter müssen gefunden und am rechten Ort ausgesprochen werden, um den Weg in das Gewölbe des Zauberers zu finden. Dort wartet ein kleiner, kostbar verzierter Gegenstand, und dann blitzt ein Funke durchs Gehirn. In Windeseile läuft der Gefangene durch Räume, die er inzwischen besser kennt als seine Sozialwohnung und dann:



Galahad: Eine rätselhafte Burg

Der Heimweg. AHHHH! Das Spiel verabschiedet sich mit dem Text: „Ich fühle mich schon viel besser! Endlich befreit aus dem bedrohlichen Schloß und vom bösen Zauberer! Die Vögel singen, der Himmel zeigt sein wolkenloses Blau, und Sie sind ein Abenteurer!"

Ich fühle mich geschmeichelt und nach all den unglaublichen Rätselqualen belohnt und erleichtert. Nur so aus Gewohnheit gebe ich ein N für die Bewegung nach Norden ein, und dieser unglaublich böse Computer antwortet: „Hier können Sie nicht weitergehen." Ich tippe O, W, S, R, H und immer wieder die gleiche Antwort: „Hier können Sie nicht weitergehen". Ist das die langersehnte Freiheit, für die ich so viele Nächte über Rätsel eines geisteskranken Zauberers gebrütet habe?

Alfred Görgens/Karl-Heinz Koch

Scan: S. Höltgen

gabe-Ports arbeiten will. Beim Entwickeln von kommerziellen Programmen wichtig ist die Anweisung PRINT USING. Damit ist es möglich, Zahlenwerte in einem bestimmten Format auszugeben, beispielsweise mit drei Stellen vor dem Komma und zwei Stellen dahinter. Soll in einem Programm eine bestimmte Befehlsfolge öfter vorkommen, kann diese mit DEF FN definiert werden. Etwa die Berechnung der Mehrwertsteuer einer Zahl. Man definiert die Funktion MEHR (ZAHL) als ZAHL=ZAHL+0.14. Gibt man jetzt die Anweisung X=MEHR(100), so enthält die Variable X anschließend den Wert 114.

Programmfehler im Griff

In jedem Fall ärgerlich sind Fehler im Programmablauf. Einigen Komfort bietet hier der BASIC-Interpreter des CPC464. Tritt ein nichterlaubter Zustand ein, bleibt der Rechner stehen und zeigt die Zeile an, in welcher der Fehler aufgetreten ist.

Eine bereits weitverbreitete Möglichkeit, Programme zu testen, ist die TRON-Anweisung. TRON bedeutet TRACE ON. Der Rechner gibt die Zeilennummer aus, die gerade bearbeitet wird. So kann der Weg des Interpreters durch das Programm mühelos verfolgt werden. Eine weitere Art, Fehler zu beheben, die vor allem bei Personal-Computern vorkommt, ist die Anweisung ON ERROR. Der Programmierer legt fest, wie das Programm im Falle eines Fehlers reagieren soll. Dazu lassen sich mit ERR der Fehler-Code und mit ERL die Zeilennummer abfragen.

Obwohl das Auffinden von Fehlern relativ rasch erfolgt, ist doch das Beheben beim PC464 nicht nach dem neuesten Stand. Da kann nämlich mit dem Cursor eine zu ändernde Stelle angefahren und Zeichen eingefügt beziehungsweise gelöscht werden. Dieses Verfahren, Bildschirm-Editor genannt, besitzt der Rechner nicht. Hier gibt es noch das etwas veraltete System des Zeilen-Editors: Die zu ändernde Programmzeile muß vorher mit EDIT aufgerufen werden.

Bei vergleichbaren Rechnern kaum vorhanden ist die komfortable Programmierung der vier internen Zeitgeber. So kann man mit dem Befehl AFTER erreichen, daß die Ausführung des folgenden Befehls erst nach einer vorgegebenen Zeit ausgeführt wird. Eine weitere Möglichkeit, den Timer zu nutzen, ist die EVERY-Anweisung. Damit lassen sich Befehle in bestimmten Zeitintervallen wiederholen. Außerdem kann man über eine Variable, TIME genannt, die Zeit ermitteln, die seit dem Einschalten des Rechners vergangen ist.

Im Preis inbegriffen ist ein Monitor mit mittlerer Auflösung. Dies hat den Vorteil, daß der Rechner ohne eigenes Netzteil auskommt. Die Spannungsversorgung erfolgt über ein Verbindungskabel zum Monitor. Wer jedoch ein Fernsehgerät verwenden will, kann den Rechner auch solo mit einem Zusatzmodulator erwerben. Dort sind ebenfalls das notwendige Netzteil und ein RGB-Ausgang untergebracht.

Die Programmierung des Bildschirms erfolgt in drei Betriebsarten. Modus 0 mit 25 Zeilen und jeweils 20 Zeichen. Die Grafikauflösung beträgt 160 × 200 Bildpunkte, und 16 der 26 vorhandenen Farben können gleichzeitig dargestellt werden. Auf 40 Zeichen pro Zeile kommt der Modus 1 mit einer Grafikauflösung von 320 × 200 Punkten. Die Zahl der gleichzeitig darstellbaren Farben geht auf vier zurück. Nur noch zwei verschiedene Farben, dafür aber 80 Zeichen pro Zeile und eine hervorragende Auflösung von 640 × 200 Punkten bietet Modus 2. Der Anwender ist somit in der Lage, zwischen vielen Farben und hoher Grafikauflösung selbst zu entscheiden. Der Bildschirmaufbau ist analog zum Sinclair Spectrum: Es gibt einen Bildschirmrand (Border), die Schreibfläche (Paper) und die Schriftfarbe (Ink). Besonders gut ist die Möglichkeit, bis zu acht Bildschirmfenster zu definieren. Diese beliebig großen Ausschnitte werden genauso wie der gesamte Bildschirm angesprochen. Die erzeugten Grafiken können zwar mit Text gemischt werden, die Programmierung bleibt jedoch etwas zurück. Lediglich Linien lassen sich per Befehl zeichnen; für Kreise oder Rechtecke muß sich der Anwender eigene Routinen ausdenken.

Monitor optimal integriert

Der Entwurf eigener Zeichen ist ziemlich problemlos. Dies ist vor allem dann ein enormer Vorteil, wenn demnächst auch von Schneider ein Matrixdrucker auf den Markt kommt, der in der Lage ist, alle Zeichen des Rechners abzubilden. Der Anwender kann außerdem den vorhandenen Zeichensatz sowie die Tastenbelegung jederzeit ändern. Von Haus aus sind beim CPC464 keine Funktionstasten vorhanden. Per Befehl läßt sich jedoch die separate Zehnertastatur umprogrammieren, so daß auf Druck die Eingabe einer gesamten Zeichenfolge möglich ist. Außergewöhnlich gut gelöst bei der Schreibmaschinentastatur ist die Bedienung: Sowohl die Ansprechzeit als auch die Frequenz der automatischen Tastenwiederholung sind nahezu stufenlos regelbar.

Ton mit Stereoeffekt

Zur Erzeugung von Geräuschen und Tönen stehen auf dem CPC464 drei Tongeneratoren zur Verfügung. Eine Ausgangsbuchse für die heimische Stereoanlage ist vorhanden. Die Programmierung ist einfach: Tongenerator A versorgt den linken Stereokanal, B ist für beide Kanäle und Generator C ertönt aus dem rechten Kanal. Im Rechner eingebaut ist ein Monolautsprecher mit regelbarer Lautstärke. Ebenfalls im Rechner integriert ist ein Kassettenrecorder. Die Datenrate von 1000 Baud läßt sich im Speedload-Modus verdoppeln.

An Schnittstellen bietet der CPC464 noch eine parallele Centronics-Schnittstelle zum Anschluß eines Druckers sowie ein Joystick-Interface, an den auch die atari-kompatiblen Steuerknüppel passen. Für zwei Spieler muß der Schneider-Joystick verwendet werden, der das Port für den zweiten Joystick besitzt.

Eine Kampfansage an Commodore oder Atari ist neben den hervorragenden Qualitäten des CPC464 auch sein Preis: Rund 900 Mark inklusive Schwarzweiß-Monitor und knapp 1400 Mark mit dem Farbmonitor. 50 fertige Programme auf Kassette aus den Bereichen Textverarbeitung, Kalkulation, Spiele, Programmiersprachen und Lernen (alles in deutscher Sprache) sind ebenfalls vorhanden. Ende des Jahres folgt eine 3-Zoll-Diskettenstation mit 160 KByte Speicherkapazität, dem Betriebssystem CP/M und der Programmiersprache Logo sowie eine serielle Schnittstelle. Man darf gespannt sein, wie sich dieser außergewöhnlich gute Rechner hierzulande durchsetzt. *– wt*

Vor- und Nachteile

+ Umfangreiches BASIC
+ Schnelle Bildschirmausgabe
+ Separate Zehner-Tastatur
+ Günstiger Preis mit Monitor
+ Eingebauter Recorder
+ Programmierbare Timer
– Bisher keine Software
– Kein Bildschirm-Editor
– Kaum Grafikbefehle

Scan: S. Höltgen

Kurven-Plot

Stetige Funktionen können in einem gewünschten Intervall geplottet und deren Nullstellen bestimmt werden

Das Programm wählt zum Plotten automatisch die größtmögliche Auflösung und Genauigkeit.

Nach der Darstellung kann das Programm sofort beendet werden, oder es können die Nullstellen mittels Iteration (Regula Falsi) im Intervall berechnet werden. Dabei gibt der Home-Computer nur Nullstellen mit Vorzeichenwechsel aus. Falls Nullstellen im Koordinatensprung auftreten, können diese doppelt aufgeführt sein.

Das Programm beginnt mit einer ausführlichen Bedienungsanleitung.

Dipl.-Ing. J. Seidel

```
100 REM     KURVENDISKUSSION
110 REM     ----------------
120 REM
130 REM  FUNKTION MUSS IN
         ZEILE 200
         DEFINIERT WERDE
N.
140 REM
150 REM
160 REM
170 CALL CLEAR
180 DIM Y(32)
190 PRINT "***** KURVENDISKU
SSION *****"::
200 DEF F(X)=SIN(X)+3/SIN(X)
210 PRINT "DIE ZU UNTERSUCHE
NDE FUNK-  TION MUSS IN ZEIL
E 200      DEFINIERT WER
DEN!"::
220 PRINT "DAS PROGRAMM STEL
LT DEN     GRAPH EINER STETI
GEN FUNK-  TION DAR UND
BERECHNET DIE  NULLSTELLEN."
::
230 PRINT "NACH DEM PLOT DES
 GRAPHEN   BESTEHT WAHLMOEGL
ICHKEIT:   "::
240 PRINT "TASTE {N} => DIE
NULLSTEL-  LEN IM INTERVALL
WERDEN     BERECHNET."::
250 PRINT "TASTE {E} => PROG
RAMMENDE."::
260 PRINT "START MIT BELIEBI
GER        TASTE!":
270 CALL KEY(0,K,S)
280 IF S=0 THEN 270
290 CALL CLEAR
300 PRINT "IN WELCHEM INTERV
ALL DER    ABSZISSE SOLL DER
 GRAPH     DARGESTELLT W
ERDEN?"::
310 PRINT "untere Intervallg
renze:":::
320 INPUT "X= ":XU
330 PRINT :::"obere Interval
lgrenze:":::
340 INPUT "X= ":XO
350 INC=ABS(XO-XU)/30
360 CALL CLEAR
370 X=XU
380 REM   FELD Y WIRD MIT 31
          FUNKTIONSWERTEN
          BESETZT
390 REM   -----------------
400 PRINT TAB(5);"FUNKTIONSW
ERTE:":"    ---------------"
::::
410 FOR I=2 TO 32
420 Y(I)=F(X)
430 PRINT TAB(5);"F(X)= ";Y(
I)::
440 X=X+INC
450 NEXT I
460 CALL CLEAR
470 PRINT "PLOT ERFOLGT MIT
FOLGENDEN  MASSTAEBEN:":"---
-----------------------
"::::::
480 PRINT "X-INCREMENT= ";IN
C::::
490 REM   BESTIMMUNG DES
500 REM   GROESTEN POSIT.
510 REM   ORDINATENWERTES.
520 REM   --------------
530 PMAX=0
540 FOR I=2 TO 32
550 IF Y(I)<=PMAX THEN 570
560 PMAX=Y(I)
570 NEXT I
580 REM    BESTIMMUNG DES
590 REM    KLEINSTEN NEGAT.
600 REM    ORDINATENWERTES
610 REM    ---------------
620 NMAX=0
630 FOR I=2 TO 32
640 IF Y(I)>=NMAX THEN 660
650 NMAX=Y(I)
660 NEXT I
670 IF ABS(NMAX)>ABS(PMAX)TH
EN 700
680 FINC=ABS(PMAX)/11
690 GO TO 710
700 FINC=ABS(NMAX)/11
710 PRINT "Y-INCREMENT= ";FI
NC::::
720 GO SUB 1630
730 CALL CLEAR
740 CALL HCHAR(13,3,45,29)
750 CALL HCHAR(13,31,62)
760 CALL HCHAR(13,32,120)
770 CALL HCHAR(13,2,60)
780 GO SUB 1370
790 FOR I=2 TO 32
800 GO SUB 1190
810 Z=13-RVAL
820 CALL HCHAR(Z,I,42)
830 NEXT I
840 REM  SELECT
850 REM  ------
860 REM  N:NULLSTELLEN
870 REM  E:ENDE
880 REM
890 CALL KEY(0,K,S)
900 IF S=0 THEN 890
910 IF K<69 THEN 890
920 IF K>78 THEN 890
930 IF K=69 THEN 1100
940 IF K<78 THEN 890
950 REM  NULLSTELLENSUCHE
960 REM  ----------------
970 CALL CLEAR
980 PRINT TAB(9);"NULLSTELLE
N:":TAB(9);"-------------":::
990 INC=(XO-XU)/100
1000 XJ=XU
1010 XI=XU+INC
1020 FOR I=1 TO 100
1030 IF SGN(F(XI))=SGN(F(XJ)
)THEN 1050
1040 GO SUB 1750
1050 XJ=XJ+INC
1060 XI=XI+INC
1070 NEXT I
1080 GO SUB 1630
1090 GO TO 1110
1100 CALL CLEAR
1110 END
1120 REM
1130 REM
1140 REM
1150 REM
1160 REM
1170 REM
1180 REM
1190 REM  SUB ROUND INTEGER
1200 REM  -----------------
1210 REM   VALUE:REAL-ZAHL
1220 REM  RVAL:GERUNDETE ZAH
L
1230 REM
1240 VALUE=Y(I)/FINC
1250 IVAL=INT(VALUE)
1260 IF VALUE-IVAL>=.5 THEN
1290
1270 RVAL=IVAL
1280 GO TO 1300
1290 RVAL=IVAL+1
1300 RETURN
1310 REM
1320 REM
1330 REM
1340 REM
1350 REM
1360 REM
1370 REM    SUB SUCHE DES
1380 REM    KOORDINATEN -
1390 REM    URSPRUNGS
1400 REM    ------------
1410 REM
1420 IF XU>0 THEN 1560
1430 X=XU
1440 FOR I=2 TO 32
1450 IF X>=0 THEN 1490
1460 X=X+INC
1470 NEXT I
1480 GO TO 1560
1490 IF ABS(X)<ABS(X-INC)THE
N 1510
1500 I=I-1
1510 CALL VCHAR(1,I,124,24)
1520 CALL HCHAR(13,I,43)
1530 CALL HCHAR(2,I,94)
1540 CALL HCHAR(1,I,121)
1550 CALL HCHAR(24,I,118)
1560 RETURN
1570 REM
1580 REM
1590 REM
1600 REM
1610 REM
1620 REM
1630 REM  SUB TIME DELAY
1640 REM  -------------
1650 REM
1660 FOR DELAY=1 TO 3000
1670 NEXT DELAY
1680 RETURN
1690 REM
1700 REM
1710 REM
1720 REM
1730 REM
1740 REM
1750 REM   SUB REGULA FALSI
1760 REM   ---------------
1770 REM
1780 REM  X1,X2:STARTWERTE
1790 REM  EPS:FEHLER
1800 REM
1810 IT=0
1820 X1=XI
1830 X2=XJ
1840 EPS=1E-6
1850 Y1=F(X1)
1860 Y2=F(X2)
1870 IF ABS(X1-X2)<1E-99 THE
N 1960
1880 M=(Y1-Y2)/(X1-X2)
1890 IF M=1E-20 THEN 2020
1900 XNEU=X2-Y2/M
1910 X1=X2
1920 X2=XNEU
1930 IF IT>100 THEN 1990
1940 IT=IT+1
1950 IF ABS(Y2)>EPS THEN 185
0
1960 PRINT TAB(9);"X= ";X2:
1970 PRINT TAB(6);"F(X)= ";Y
2::
1980 GO TO 2020
1990 CALL SOUND(100,110,2)
2000 PRINT ::"KEINE KONVERGE
NZ BEI DIESER NULLSTELLE!"::
2010 PRINT "ICH SUCHE WEITER
!"::
2020 RETURN
```

Scan: S. Höltgen

Computer im Kreuzfeuer

Halbblind, impotent und leicht bescheuert schleppt er sich durch sein vereinsamtes Dasein. Verlassen von Frau und Freundin, kämpft er vergebens gegen seine nekrophile Veranlagung an, gemieden von der normalen Bevölkerung. Ein phantasieloser Langeweiler, süchtig und pervers dazu: der Computer-Freak, wie er leibt und darbt.

Das arme Schwein ist unversehens in heftiges Kreuzfeuer geraten, Fachleute jeglicher Couleur – vor allem aus der psychologischen Ecke – nehmen ihn erbarmungslos aufs Korn und bis in die letzten Einzelheiten auseinander. Übrig bleibt eine rundum deformierte Persönlichkeit, Frankensteins Monster oder einem veritablen Zombie ähnlicher als dem durchschnittlichen Zeitgenossen.

Die neue Droge

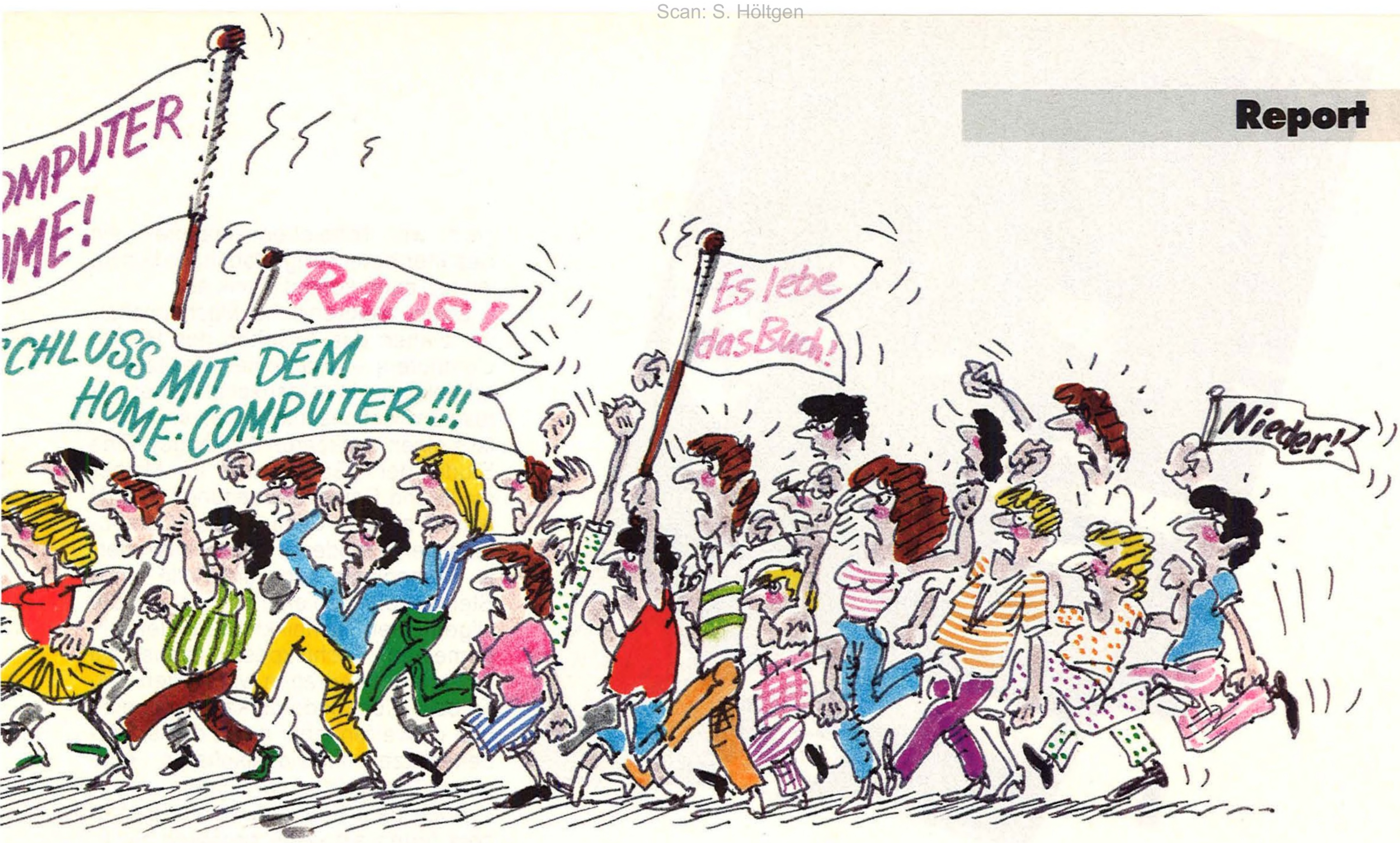
Was war geschehen? Eine neue Technologie tauchte unversehens auf, und eine Menge meist jüngerer Leute ließ sich begeistert darauf ein. Das ist verdächtig, das riecht nach Anarchie, das darf nicht sein – sagten sich die geistigen Vorturner der Nation und tauchten die Feder ins Giftfaß. Die neue Droge namens Home-Computer rief zugleich die übelsten Assoziationen zu großen Brüdern, Überwachungsapparaten, Arbeitsplatzvernichtern und vollautomatischen Kriegsauslösern wach. Die User verstehen nur noch Bahnhof, schien ihnen das Schlimmste an ihrem Computer doch nur das Design und das Manual zu sein.

Angesichts der Flut von Verdächtigungen, Mutmaßungen und Analysen, die über die verdatterten Computer-Fans hereinbricht, scheint die Zeit für einen Gegenangriff gekommen. Mal sehen, was die Argumente der Kritikaster hergeben.

Computer machen krank, stören die geistige Entwicklung und ramponieren soziale Bindungen – so die Thesen der Mahner und Warner. Der Autor Jochen Köhler hat – in der Zeitschrift „päd. extra" – den „Volkscomputer auf dem Vormarsch in die Kinderzimmer" geortet, eine „epidemisch sich ausbreitende Sucht" erzeugend: „Dabei zeigen die Sprößlinge durch die Bank Symptome innerer Unruhe, von Interesselosigkeit für Anderweitiges und Störanfälligkeit durch alle möglichen Bagatellen." Klassische Suchtsymptome glaubt auch Dr. Thomas von Randow („Die Zeit") in der Computer-Faszination entdeckt zu haben: „Schüler, die ihr unterliegen, vernachlässigen völlig die anderen Fächer... es muß dringend untersucht werden, welche Möglichkeiten die Schule besitzt, dem ‚Hackertum' gegenzusteuern." Diese Beobachtungen, sofern sie zutreffen, lassen sich nahtlos auf jedes andere Hobby – vom Fußball bis zum Schachspiel – übertragen. Anders dagegen die ständige Konfrontation mit der flimmernden Mattscheibe: Sie kann „zu spezifischen Beschwerden wie Augenschmerzen, Streß, Erschöpfung, Rükkenschmerzen, Kopfschmerzen und Hautentzündungen führen" – so John Evans in seinem Bericht an den „Club of Rome" (1982). Harmlose Wehwehchen gegen die „verheerenden Auswirkungen auf das Sexualleben", welche die Londoner „Gesellschaft für Gleichberechtigung" bei Terminal-Benutzern ausgemacht haben will: „Eine der Streßwirkungen ist es, Impotenz bei Männern und Mangel an sexuellem Interesse bei Frauen zu verursachen."

Die Schreckensvision verliert jedoch schnell an Gewicht, da der echte Freak seinen Rechner jeder Zweierki-

Scan: S. Höltgen



Die Opposition macht mobil: Kritiker nehmen den Home-Computer aufs Korn – und seine Benutzer dazu. Steht der Untergang des Abendlandes wieder mal bevor – diesmal rechnergestützt?

ste vorzieht. Nach einer „Studie" der Eheberaterin Jean Holland (Los Angeles) bringt der Home-Computer jede Ehe in Gefahr, wenn nicht zum Scheitern. Innerhalb eines Beobachtungszeitraumes von fünf Jahren gingen 50 Prozent der computer-beglückten Ehen in die Brüche, die andere Hälfte ließ sich scheiden – vom Rechner. Die findige Psychologin fand auch gleich heraus, weshalb die Maschinchen so gefährlich sind: Sie geben sich stets ansprechbar, haben keine Launen, sagen nie nein und führen alle Befehle und Wünsche des Mannes sofort aus.

Totale Identifikation

Aber nicht nur die Ehefrau ergreift die Flucht, der neue Hausgenosse vertreibt gar die gesamte Freundesschar: „Der Computer wird zum Selbstzweck, das fortwährende Programmieren zur neuen Spielleidenschaft, indem man fleißig strebend sich bemüht, stets neue Programmiertricks einzuflechten. Die totale Identifikation mit dem Computer und die damit allzuoft einhergehende soziale Abschottung bis hin zur Unfähigkeit zwischenmenschlicher Kommunikation", sieht Johannes Dünnwald in dem Buch „Schöne elektronische Welt" (Rowohlt 1982) heraufdämmern. Letztlich „wird das Gerät als Partner akzeptiert und die Struktur der Interaktion von der Maschine bestimmt", prophezeien die Autoren des Werkes „Maschinen-Menschen – Mensch-Maschinen" (Rowohlt 1983).

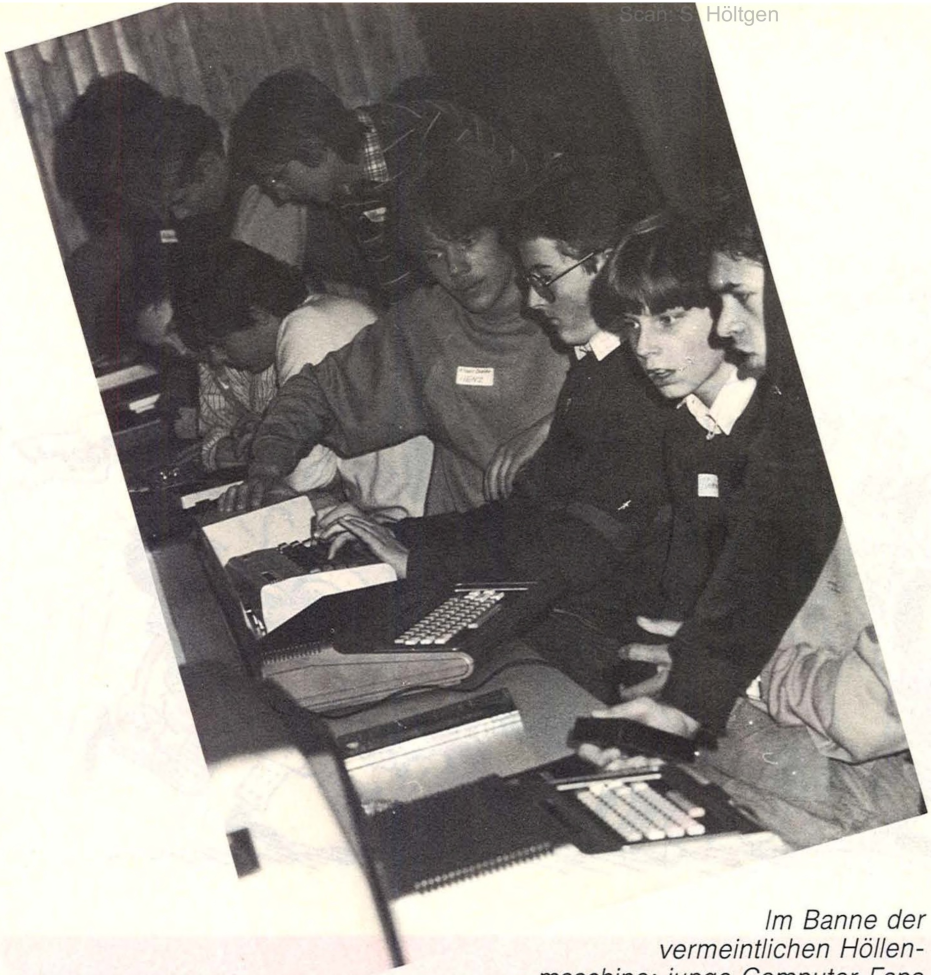
Liebesgeflüster demnächst nur noch auf BASIC? Die Diskette als Lebensgefährtin? Es kommt noch schlimmer: Allen Ernstes behauptet eine Reihe ehrenwerter Zeitgenossen, daß der ständige Umgang mit dem Home-Computer die Psyche seines Besitzers irreparabel verbiege: „Die als Verhaltensroutine in der Psychostruktur repräsentierte Maschine ist komplementär zu der außen befindlichen Hard- und Software beziehungsweise zu den Verhaltensroutinen der anderen an der Situation beteiligten Personen", stellt einer der „Maschinenmenschen"-Autoren fest, und „... gerade gegenwärtig wird in aller Deutlichkeit sichtbar, wie durch lebensgeschichtlich frühe Sozialisationsprozesse die Psychostruktur automatengerecht geformt wird".

Alles klar? Der zwangsläufig psychostrukturgeschädigte HC-Leser hat natürlich sofort verstanden, was ihm fehlt – sein Innenleben gleicht sich immer mehr einer Z 80-CPU an. „Durch dieses ständige Anpassen verlieren wir allmählich unsere menschliche Eigenständigkeit im Denken wie im Handeln, die Computer-Wirklichkeit wird unsere Wirklichkeit",folgert Norbert R. Müller („Schöne elektronische Welt") und „Am Ende steht das computerisierte Leben, der ‚automatisierte Mensch'..." Der interessiert sich nicht mehr – nach Erich Fromm – für andere Menschen, Natur und lebendige Strukturen, sondern allenfalls für mechanische nichtlebendige Kunstprodukte. Dieses nekrophil veranlagte Ekel zeichnet sich vor allem durch seine „narzistischen, schizoiden und autistischen Eigenschaften" aus, sowie durch die Spaltung von Denken und Fühlen.

Digitales Denken

Die Warnung, daß zu innige Beschäftigung mit dem Computer letztlich zu etwas eigenartigen Denkprozessen führe, findet sich häufig. „Die Schulung des Geistes mit auch noch so raffinierten Computer-Programmen bringt stets eine einseitige Orientierung an formallogischen Strukturen

Scan: S. Höltgen



Im Banne der vermeintlichen Höllenmaschine: junge Computer-Fans

mit sich", stellt der bereits zitierte Jochen Köhler fest. Zu einem ähnlichen Ergebnis kommt Dünnwald. Er meint, „daß die zunehmend starre Fixierung auf Computerleistung ... zur blinden Konditionierung des Menschen auf ausschließlich meß- und quantifizierbare, damit aber computergerechte Phänomene unserer Mitwelt ... führt". Kreativität, Phantasie, Emotionen, Spontaneität und Kommunikation bleiben da auf der Strecke.

Nach Werner Jakobsmeier („Dokument und Analyse"), sehen amerikanische Wissenschaftler die individuelle Entfaltung der Persönlichkeit durch das Computer-Hobby gefährdet: „Nach ihrer Einschätzung ist der Mensch der Computer-Gesellschaft nur noch gewohnt, in digitalen Schritten zu denken. Computer-Arbeit zwinge zur völlig einseitigen Ausprägung nur einer Gehirnhälfte."

Angst macht sich breit

Und mit der ist es auch nicht allzuweit her. Prof. Ernst Becker (TH Darmstadt) befürchtet, daß der ständige Umgang mit dem Computer „die Fähigkeit der geistigen Durchdringung von technischen Problemen" verdränge. Außerdem verführe er zu Mißbrauch aus Bequemlichkeit, Denkfaulheit und Ideenarmut. Den harschen Argumenten vieler Wissenschaftler entspricht die Stimmungslage in der Bevölkerung: Eine repräsentative Umfrage der Wickert-Institute ergab, daß fast die Hälfte (49 Prozent) der erwachsenen Bundesbürger Angst vor Computern empfindet.

So weit, so schlecht: Den Computer-Fans weht ein schneidend kalter Wind ins Gesicht, um das Image ihrer Lieblingsbeschäftigung ist es außerordentlich schlecht bestellt. Experten – und solche, die sich dafür halten – bemühen sich eifrig, das Meinungsklima auf den absoluten Nullpunkt abzusenken, mit beträchtlichem Erfolg.

Wünschenswert wäre jetzt die Rückkehr auf eine emotionsfreie Diskussionsbasis, die tatsächliche Vor- und Nachteile der neuen Technologie gegeneinander abwägt. Tatsache ist, daß viele auch ernst zu nehmende Kritiker das Gefahrenpotential, das die Groß-EDV in sich birgt, undifferenziert auf die wehrlosen Heimgeräte übertragen: Die großen Datensammlungen laden geradezu zum Mißbrauch ein, die Sammel- und Verknüpfungswut einiger Behörden hat sich inzwischen herumgesprochen. Nicht umsonst haben die amtlichen Datenschützer in der Bundesrepublik einen schweren Stand, der individuelle Freiraum des Bürgers ist einer steigenden Gefährdung ausgesetzt. Ihren miesen Ruf verdankt die Computer-Technologie auch dem verbreiteten Einsatz als Rationalisierungsinstrument, sei es in Büros, sei es in Fabriken – Arbeitsplätze stehen auf dem Spiel. Wenn dann sogar noch in so teuflischen Dingern wie Mittelstreckenraketen ein Rechner bereitwillig Pfadfinderdienste leistet, dann kommt Panik auf.

Derlei elektronische Machenschaften ziehen das Ansehen des Home-Computers – und seines Besitzers – erheblich in Mitleidenschaft. Eher diffuse Zivilisationskritik findet hier ebenfalls einen dankbaren Aufhänger: Eine Studie der „Gesellschaft für Mathematik und Datenverarbeitung" (GMD) kommt zu dem Schluß, „daß Computer als eine moderne und vieldiskutierte Technologie auch die Rolle eines Stellvertreters spielen und viel von der allgemeinen Technikkritik und dem allgemeinen Technikpessimismus auf Computer übertragen wird". Den Ataris oder Commodores kann man einiges vorwerfen, aber nicht, daß sie Arbeitsplätze, Individualsphären oder ganze Landstriche vernichten.

Auch dem Kleinhirn seines Benutzers fügen sie keine größeren Schäden zu – bei vernünftigem Gebrauch beeinträchtigen sie das Denkvermögen ebensowenig wie die Gesundheit. Der stimmgewaltige Chor der Pädagogen und Kulturkritiker lief noch jedes Mal zu Hochform auf, wenn ein neues Unterhaltungsmedium auftauchte – sei es der Tonfilm oder das Fernsehen. Wie die Geschichte zeigt, hat die robuste Bevölkerung auch diese Heimsuchungen ohne sichtbare Mängel an Leib und Seele überstanden.

Bange machen gilt nicht

Die vielbeschworene Einengung der Denkvorgänge auf ein bestimmtes logisches System verliert ebenfalls ihre Schrecken, wenn man sich vergegenwärtigt, wie steinalt der Streitpunkt schon ist: Die Positivismus-Diskussion brodelt seit der Mitte des 19. Jahrhunderts vor sich hin (Positivisten nehmen nur harte Tatsachen zur Kenntnis; Spekulationen jeglicher Art lehnen sie ab – die Computer-Logik muß sich ebenfalls auf meßbare Größen beschränken).

Die Vorteile der intensiven Beschäftigung mit dem Mikro-Computer lassen sich klar definieren: Wer erst mal aus eigener Erfahrung Bescheid weiß, wie ein Rechner funktioniert und welche Möglichkeiten in der Datenverarbeitung stecken, der findet sich auch in einer Welt zurecht, die derzeit von neuen Technologien gründlich umgekrempelt wird. Bange machen gilt nicht: Das beste Mittel gegen Ängste ist der Erwerb von aktuellem Wissen. Mit dem Computer geht es ganz einfach. Und Spaß macht es auch. *– hs*

Scan: S. Höltgen

zeichnen, so muß man das Programm mit STOP beenden; zum Beispiel 2000 STOP. Nach dem Programmstop kann man nun mit X=..:Y=..:GOSUB PLOT bzw. GOSUB DRAW über das Textfenster IM Direkt-Eingabemodus zeichnen.

Dieses Verfahren zur Bildschirmerweiterung ist nach einigen Programmänderungen auch bei anderen Grafikbereichen anwendbar. Die Blockaufteilung und die etwas umständlichen Plot- und Drawto-Verfahren, können dabei meist entfallen. *Wolfgang Kress*

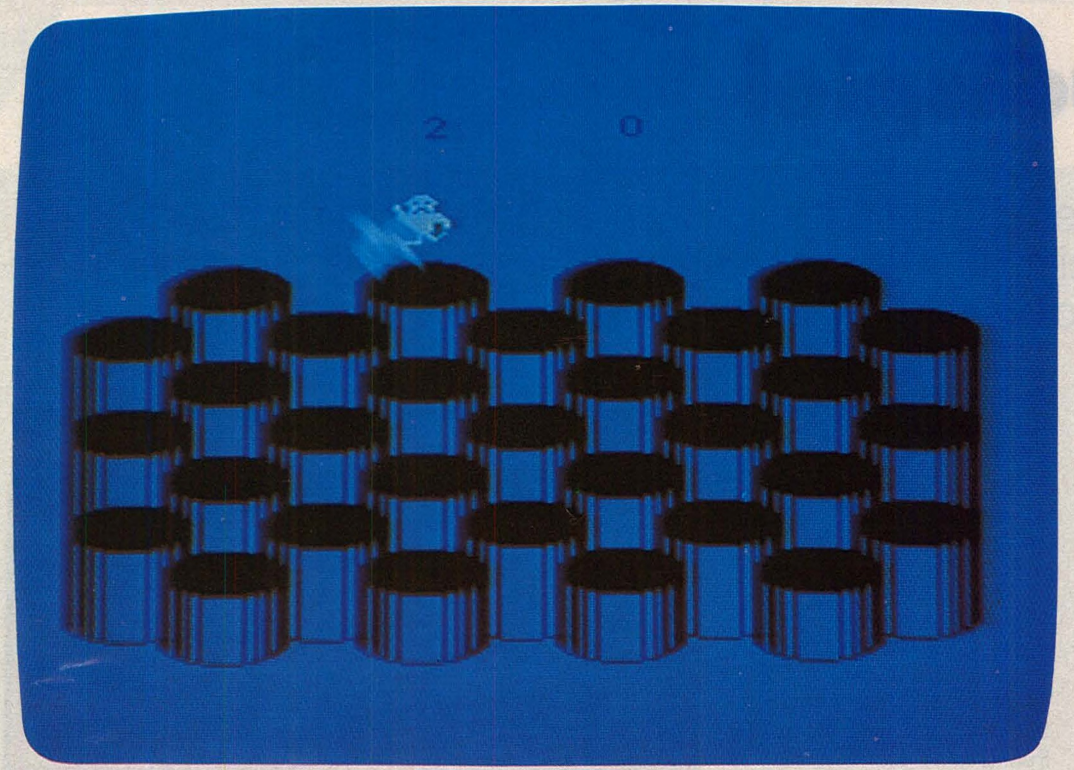
```
660 ? #6;CHR$(125)
670 POKE 88,BAAL:POKE 89,BAAH
699 REM . DLI-ZEIGER SETZEN
700 POKE 560,DLNL:POKE 561,DLNH
749 REM . BILDSCHIRM IST FERTIG
750 GOTO 2000
800 REM .
999 REM . SUB PLOT
1000 IF Y>46 THEN PLOT X,Y-47
1010 IF Y<47 THEN GOSUB UP:PLOT X,Y:GO
SUB DOWN
1020 YA=Y:XA=X
1030 RETURN
1099 REM .SUB DRAW
1100 IF YA<47 AND Y>46 THEN YE=47:GOSU
B UP:GOSUB TEIL:GOSUB ZWI:PLOT XE,0:DR
AWTO X,Y-46:GOSUB DOWN:PLOT X,Y-47
1200 IF YA>46 AND Y<47 THEN YE=0:GOSUB
 ZWI:PLOT XA,YA-46:GOSUB TEIL:GOSUB UP
:PLOT XE,47:DRAWTO X,Y:GOSUB DOWN
1300 IF YA>46 AND Y>46 THEN DRAWTO X,Y
-47
1400 IF YA<47 AND Y<47 THEN GOSUB UP:D
RAWTO X,Y:GOSUB DOWN
1410 XA=X:YA=Y
1420 RETURN
1499 REM .SUB TEIL
1500 V=(X-XA)/(Y-YA)
1510 XE=XA+V*(46-YA+1*(YE=47))
1520 DRAWTO XE,YE
1530 XE=X-V*(Y-46+1*(YE=0))
1540 RETURN
1599 REM .SUB UP
1600 POKE 88,BANL:POKE 89,BANH
1610 RETURN
1699 REM .SUB DOWN
1700 POKE 88,BAAL:POKE 89,BAAH
1710 RETURN
1799 REM .SUB ZWI
1800 POKE 88,BAZL:POKE 89,BAZH
1810 RETURN
1998 REM
1999 REM Kleines Demonstrationsbeispie
l
2000 SETCOLOR 4,0,6
2010 DEG :YO=119
2020 X=0:Y=YO:GOSUB PLOT
2030 C=1
2040 FOR X=0 TO 319 STEP 5
2050 COLOR C:GOSUB DRAW:C=1-C
2060 NEXT X
2070 X=159:Y=0:GOSUB PLOT
2080 FOR Y=0 TO 239 STEP 5
2090 COLOR C:GOSUB DRAW:C=1-C
2100 NEXT Y
2110 COLOR 1
2120 X=0:Y=YO:GOSUB PLOT
2130 FOR X=0 TO 319
2140 Y=119*SIN(1.128*X*2)
2150 Y=YO-Y
2160 GOSUB DRAW
2170 NEXT X
2180 GOTO 2180
```

Tonnenhopper

Zwischen einem System von Tonnen muß der Hopper herumspringen und geheime unterirdische Gänge erkunden. Nur unter einer Tonne läßt der TI-99/4A mit Extended-BASIC ein Tor zur Freiheit offen

Jeweils zwei Tonnen eines Systems sind untereinander durch einen geheimen Gang verbunden. Ziel des Spieles ist es, einmal durch jeden Gang zu gelangen und zuletzt durch die „Futschtonne" in die Freiheit. Für jeden gefundenen Gang gibt es 200 Punkte. Falls man in eine Tonne springt, die man schon einmal in irgendeiner Richtung benutzt hat, wird man wieder herauskatapultiert und bekommt außerdem 50 Punkte abgezogen.

```
100 CALL CLEAR :: OPTION BASE 1 :: DIM R
(32,2):: CALL MAGNIFY(3)
110 CALL CHAR(35,"3C4299A1A199423C")
120 DISPLAY AT(3,9):"TONNENHOPPER" :: DI
SPLAY AT(7,1):"# 1984  -  ANDREAS EHLERD
ING" :: DISPLAY AT(14,7):"VERSION ?  (0-
9)" :: ACCEPT AT(16,14)VALIDATE(DIGIT)BE
EP SIZE(-1):P
130 FOR A=1 TO P :: B=RND :: NEXT A :: C
ALL SCREEN(2):: CALL CLEAR
140 REM CHARCODES DER TONNENTEILE
150 CALL CHAR(96,"0001071F3F7FFFFF1FFFFF
FFFFFFFFFFF8FFFFFFFFFFFFFFF0080E0F8FCFEFF
FF")
160 CALL CHAR(100,"B7B7B7BFBFFFFFFF3FFFF
FFFFFFFFFFFFCFFFFFFFFFFFFFFEDEDEDFDFDFFF
FFF")
170 CALL CHAR(104,"FFFF7F3F1F070100FFFFF
FFFFFFFFF1FFFFFFFFFFFFFFFF8FFFFFEFCF8E08
000")
180 CALL CHAR(108,"FFFFFFFFFFEFEDEDFFFFF
FFFFFFFFF3FFFFFFFFFFFFFFFFCFFFFFFFFFFF7B
7B7")
190 CALL CHAR(112,"EDED6D2D1D07010020202
0202020E01F04040404040407F8B7B7B6BCB8E08
000")
200 CALL CHAR(116,"EDEDEDEDEDEDEDED20202
020202020200404040404040404B7B7B7B7B7B7B
7B7")
210 CALL CHAR(120,"EDEDEFFFFFFFFFFFF3FFFF
FFFFFFFFFFFFFCFFFFFFFFFFFFFFFB7B7F7FFFFFFF
FFF")
220 REM CHARCODES FUER SPRITES
230 CALL CHAR(128,"20203FFFFFFFFFFFFFFFF
FFFFFFFFFFF0404FCFFFFFFFFFFFFFFFFFFFFFFFF
FFF")
240 CALL CHAR(132,"FFFFFF000000000000000
0000000000FFFFFF000000000000000000000000
000")
250 CALL CHAR(136,"E03F20202020202020202
020202020202007FC04040404040404040404040
404")
260 CALL CHAR(140,"E0FFFFFFFFFFFFFFFFFFF
FFFFFFFFFFF07FFFFFFFFFFFFFFFFFFFFFFFFFFF
FF")
270 CALL CHAR(40,"20202020202020200000000
0000000000404040404040404000000000000000
00")
280 CALL CHAR(44,"FFFFFFFFFFFFFFFF000000
0000000000FFFFFFFFFFFFFFFFF0000000000000
00")
```



Möwe attackiert Hopper

Zusätzlich wird man von einer gereizten Möwe attackiert und muß als echter Tierfreund natürlich versuchen, ihr auszuweichen, um sie nicht zu verletzen. Wenn dies nicht gelingt, stürzt die Möwe in eine der Tonnen, und 200 Punkte Abzug sind fällig.

Folgende Situationen führen zu einem verfrühten Ende des Spieles: Weniger als 0 Punkte, Sprung in die geheime Futschtonne, bevor alle anderen Tonnen schon benutzt worden sind. Gesteuert wird über die Tasten:

U: nach links oben
J: nach links unten
I: nach rechts oben
K: nach rechts unten

Andreas Ehlerding

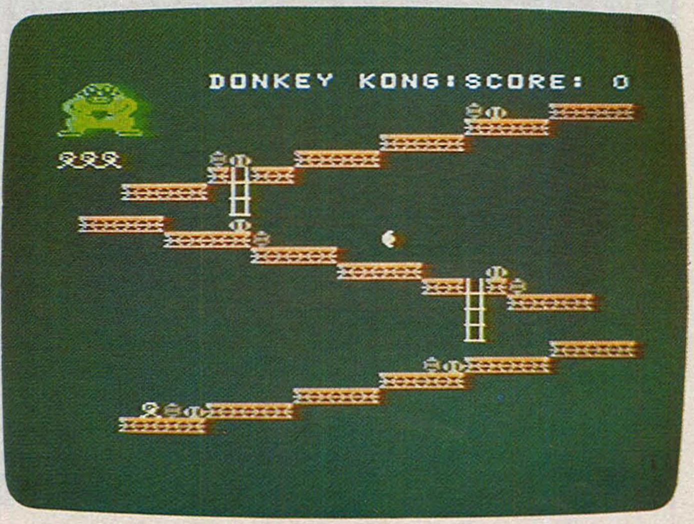
```
290 CALL CHAR(64,"03070D0705061F3F67673F
1F0301010FC0E0B0E0A060F8FCE6E6FCF8C08080
F0")
300 CALL CHAR(68,"03070D6766353F3F1F0F07
030101010FC0E0B0E060A0F8FCFEE6E6E0C46830
00")
310 CALL CHAR(72,"03070D0706051F3F7F6767
0723160C00C0E0B0E666ACFCFCF8F0E0C0808080
F0")
320 CALL CHAR(76,"03070D6766353F3F1F0F07
030101010FC0E0B0E666ACFCFCF8F0E0C0808080
F0")
330 CALL CHAR(80,"000000000001031F2FFF1F
0300000000000183870F0E0C0F0FFFEF8C0000000
00")
340 CALL CHAR(84,"0000000000031F2FFF1F03
0000000000000000030F0E0F0FFFEF8C00000000
00")
350 CALL CHAR(88,"0000000000031F2FFF1F03
0100000000000000000E0F0FFFEF8C0E0000000
00")
360 CALL CHAR(92,"00000000031F2FFF1F0301
0000000000000000C0F0FFFEF8E0F0F8783818
00")
370 CALL CHAR(60,"A078FC3E3E1F0F1F1E3E7C
7858301010 0A1F3E7FF8E0E0F0781C0C04040000
00")
380 IF LEN(A$)THEN 550
390 REM BESTIMMEN DER VERBINDUNGEN ZWISC
HEN TONNEN
400 FOR X=1 TO 32 :: IF R(X,1)THEN 420
410 R(X,1)=INT(RND*32)+1 :: IF R(R(X,1),
1)THEN 410 ELSE R(R(X,1),1)=X
420 NEXT X :: X=14
430 REM AUFBAU DER TONNEN
440 FOR H=4 TO 30 STEP 3 :: CALL VCHAR(1
0,H,117,13):: CALL VCHAR(10,H+1,118,13):
: NEXT H
450 FOR H=4 TO 22 STEP 6 :: DISPLAY AT(8
,H)SIZE(4):"`abc" :: NEXT H :: FOR V=12
TO 20 STEP 4 :: FOR H=4 TO 22 STEP 6 ::
DISPLAY AT(V,H)SIZE(4):"defg" :: NEXT H
:: NEXT V
460 FOR H=1 TO 28 STEP 6 :: FOR V=10 TO
18 STEP 4 :: DISPLAY AT(V,H)SIZE(4):"def
g" :: NEXT V :: NEXT H :: FOR H=4 TO 22
STEP 6 :: DISPLAY AT(24,H)SIZE(4):"pqrs"
 :: NEXT H
470 FOR H=4 TO 22 STEP 6 :: FOR V=9 TO 2
1 STEP 4 :: DISPLAY AT(V,H)SIZE(4):"lmno
" :: NEXT V :: NEXT H
480 FOR H=1 TO 28 STEP 6 :: FOR V=11 TO
19 STEP 4 :: DISPLAY AT(V,H)SIZE(4):"lmn
o" :: NEXT V :: NEXT H
490 FOR V=12 TO 20 STEP 4 :: CALL VCHAR(
V,3,116,2):: CALL VCHAR(V,30,119,2):: NE
XT V :: CALL VCHAR(10,3,96):: CALL VCHAR
(10,30,99)
500 CALL VCHAR(22,3,116):: CALL VCHAR(22
,30,119)
510 DISPLAY AT(23,1)SIZE(3):"pqr" :: DIS
PLAY AT(23,26)SIZE(3):"qrs" :: FOR V=14
TO 18 STEP 4 :: DISPLAY AT(V,1)SIZE(2):"
xy" :: DISPLAY AT(V,27)SIZE(2):"z{" :: N
EXT V
520 FOR H=4 TO 22 STEP 6 :: DISPLAY AT(2
3,H)SIZE(4):"tuvw" :: DISPLAY AT(22,H)SI
ZE(4):"tuvw" :: NEXT H :: FOR H=8 TO 20
STEP 6 :: DISPLAY AT(23,H)SIZE(2):"qr" :
: NEXT H
530 CALL SCREEN(5):: GOTO 560
540 REM AKTIVE FIGUREN SETZEN UND WEITER
E SPIELVORBEREITUNGEN TREFFEN
550 FOR X=1 TO 32 :: R(X,2)=0 :: NEXT X
:: PU,AZ=0 :: X=14
560 CALL SPRITE(#1,80,16,100,256,2,-12)
570 REM HOPPERROUTINE
580 IF R(X,2)THEN CALL SOUND(-50,110,2):
: PU=PU-50 :: GOTO 590 ELSE CALL SOUND(-
50,330,2):: R(X,2)=R(X,2)+1 :: AZ=AZ+2 :
: PU=PU+200 :: X=R(X,1):: IF X=32 THEN 8
30
590 IF PU<0 THEN 980 ELSE R(X,2)=R(X,2)+
1 :: DISPLAY AT(2,11)SIZE(8):USING "##
####":AZ,PU
600 V1=INT((X-.5)/4.5)+1 :: A=48*((V1/2)
-INT(V1/2)):: V=V1*16+55 :: H=(X-INT((V1
-1)*4.5)-1)*48+A+25
610 CALL POSITION(#5,OV,OH,#1,QV,QH):: D
V=OV-QV-32 :: DH=ABS(OH-QH):: M=MAX(ABS(
DV),DH):: IF M=0 THEN 630
620 CALL MOTION(#1,DV/M*12,MIN(DH/M*-12,
-6))
630 IF V<136 THEN CALL SPRITE(#3,128,2,V
+16,H,#4,132,5,V+16,H)ELSE CALL SPRITE(#
3,40,2,V+16,H,#4,44,5,V+16,H)
640 VV=-6 :: VH=0 :: CALL SPRITE(#5,136,
2,V,H,#6,140,5,V,H,#7,64,16,V,H,VV,VH)
650 K=0 :: FOR A=1 TO 3 :: FOR B=1 TO 4
:: FOR C=1 TO 2 :: CALL KEY(2,K1,S):: IF
 S THEN K=K1 :: CALL SOUND(-5,505,3)
660 NEXT C :: CALL COINC(#1,#7,12,CO)::
IF CO THEN GOSUB 890 :: REM COINCIDENCE
670 CALL PATTERN(#1,76+B*4):: NEXT B ::
NEXT A
680 RH=SGN((K/2-INT(K/2))-.2):: RV=SGN(3
.5-K):: IF K=0 THEN RV,RH=0
690 V=V+RV*16 :: H=H+RH*24 :: IF H>232 O
R H<16 OR V<64 OR V>168 THEN V=V-RV*16 :
: H=H-RH*24 :: RH,RV=0
700 IF V<136 THEN CALL SPRITE(#3,128,2,V
+16,H,#4,132,5,V+16,H)ELSE CALL SPRITE(#
3,40,2,V+16,H,#4,44,5,V+16,H)
710 CALL SPRITE(#5,136,2,V,H,#6,140,5,V,
H)
720 CALL POSITION(#7,PV,PH):: IF ABS(PH-
H)<2 THEN 750 ELSE VH=RH*((12-ABS(12-ABS
(PH-H)))/3+1):: VV=(12-ABS(PH-H))/2 :: C
ALL MOTION(#7,VV,VH)
730 CALL PATTERN(#7,68+INT(RND*3)*4):: C
ALL COINC(#1,#7,12,CO):: IF CO THEN GOSU
B 890 :: REM COINCIDENCE
740 CALL SOUND(-10,ABS(V-PV)+165,5):: GO
TO 720
750 VV=12 :: VH=0 :: CALL SPRITE(#7,68,1
6,PV,H,VV,VH)
760 CALL POSITION(#7,PV,PH):: IF PV>V-8
AND PV<176 THEN CALL DELSPRITE(#7,#6,#5,
#4,#3):: GOTO 780 ELSE CALL COINC(#1,#7,
12,CO):: IF CO THEN GOSUB 890 :: REM COI
NCIDENCE
770 CALL PATTERN(#7,68+INT(RND*3)*4):: C
ALL SOUND(-10,ABS(V-PV)+165,5):: GOTO 76
0
780 CALL SOUND(-50,165,3)
790 IF RV=-1 THEN X=X-4+SGN(RH-1)
800 IF RV=1 THEN X=X+4+SGN(RH+1)
810 GOTO 580
820 REM FUTSCH-TONNE
830 CALL DELSPRITE(ALL):: CALL CHARSET :
: IF AZ>30 THEN 870 ELSE M=2
840 DISPLAY AT(2,9):"MEHR GEDULD !" :: D
ISPLAY AT(4,3):"DAS SPIEL WAR DOCH NOCH"
 :: DISPLAY AT(6,1):"GAR NICHT ZU ENDE !
? SCHADE."
850 FOR A=1 TO 1000 :: NEXT A :: CALL HC
HAR(2,1,32,160):: DISPLAY AT(2,3):"DU HA
TTEST IMMERHIN SCHON" :: DISPLAY AT(4,8)
:STR$(PU)&" PUNKTE !!!"
860 DISPLAY AT(6,2):"PROBIERS DOCH NOCH
MAL ..." :: GOTO 1030
870 DISPLAY AT(4,8):"FABELHAFT !!!" :: F
OR A=1 TO 1000 :: NEXT A :: DISPLAY AT(6
,1):"FREUST DICH BESTIMMT. TOLL!" :: M=3
 :: GOTO 1030
880 REM COINCIDENCE
890 D=0 :: PU=PU-300 :: CALL SOUND(-50,-
7,1):: CALL MOTION(#1,0,0,#7,0,0):: CALL
 PATTERN(#1,60):: CALL SOUND(-50,-6,2)
900 CALL POSITION(#1,QV,QH,#5,OV,OH):: D
V=(OV-QV-16)/3 :: DH=(OH-QH)/3 :: CALL S
OUND(-50,-5,3):: IF INT(ABS(DH))=0 THEN
DH=SGN(DH)
910 CALL MOTION(#1,DV,DH)
920 CALL POSITION(#1,QV,QH):: IF ABS(QH-
OH)<ABS(DH)/7+1 THEN 930 ELSE CALL SOUND
(-100,(31-D)*110,D):: D=D+1 :: GOTO 920
930 IF DV<6 THEN DV=6
940 CALL DELSPRITE(#1):: CALL SPRITE(#8,
60,16,QV,OH,DV,0)
950 CALL COINC(#5,#8,ABS(DV)/7+1,CO):: I
F CO THEN 960 ELSE CALL SOUND(-100,(30-D
)*110,D):: D=D+1 :: GOTO 950
960 CALL DELSPRITE(#8):: CALL SOUND(-100
,115,1):: CALL SPRITE(#1,80,16,100,256,0
,-12):: CALL SOUND(-100,110,2):: CALL MO
TION(#7,VV,VH):: RETURN
970 REM PUNKTE UNTER "O"
980 CALL DELSPRITE(ALL):: CALL CHARSET :
: IF PU=-50 THEN M=0 ELSE M=1
990 IF M=0 THEN DISPLAY AT(2,5):"SCHREIB
 DIR DOCH DAS" :: DISPLAY AT(4,2):"NAECH
STE MAL LIEBER AUF WO" :: DISPLAY AT(6,8
):"DU SCHON WARST !"
1000 IF M=1 THEN DISPLAY AT(2,2):"IST DA
S SPIEL SO SCHWER ZU" :: DISPLAY AT(4,1)
:"VERSTEHEN ???  PARAGRAPH 1 :" :: DISPL
AY AT(6,1):"WENN VOGEL-DANN SCHNELL WEG!
"
1010 GOTO 1030
1020 REM SPIEL ?
1030 FOR A=1 TO 1000 :: NEXT A :: CALL H
CHAR(2,1,32,160):: DISPLAY AT(2,10):"NOC
H MAL ?" :: ACCEPT AT(4,14)VALIDATE(UALP
HA)BEEP SIZE(2):A$ :: CALL HCHAR(2,1,32,
160):: IF A$="JA" OR A$="J" OR A$="Y" TH
EN 230
1040 CALL CLEAR
1050 END
```

```
42020 print"███"spc(6)"      ▌      ▌ ▐███▌ ▐███▌ "spc(13);
42030 printspc(6)" ▐██▌ ▐██▌ ▌ ▐███▌ ▐███▌ "spc(13);
42040 printspc(6)" ▐██▌ ▐██▌ ▌ ▐███▌ ▐███▙"spc(13);
42050 printspc(6)" ▐██▌       ▌     ▌     ▌ "
42060 poke198,0:wait198,1:gosub44000
42070 print"████▙"chr$(142)
42080 print"▌|♦etzt koennen sie alle buchstaben-     █JK ♦ tasten betaetigen !!"
42090 poke198,0:wait198,1
42100 print"███  man kann sagen, sie sind fluegge !!"
42110 fora=0to2000:next
42120 print"█████████"
42130 print"────────────────────────────────────────"
42150 fora=2to5:pokev+38+a,a:next:pokev+41,1
42160 pokev+29,0:pokev+23,31:pokev+21,31
42170 fora=0to8step2:pokev+a,55+a*25:pokev+a+1,160:next
42180 fora=0to800:next
42190 y=160:d=1
42200 y=y-(19-y/9):ify<0then42240
42210 pokev+5,y:ifd=0thenpokev+29,4:pokev+4,145:d=1:goto42230
42220 pokev+29,0:pokev+4,155:d=0
42230 forb=0toy*1.5+100:next:goto42200
42240 fora=0to1500:next
42250 fory=0to255step2:pokev+5,y:next
42260 fora=1to32:poke53281,a:next
42270 print"██      nur uebung macht den meister !!     "
42280 fora=0to18:pokev+23,0:forb=0to120:next:pokev+23,31
42290 forb=0to300:next:next
42300 poke198,0:wait198,1:print"█"chr$(14):pokev+21,0:gosub10100:return
42998 :
42999 :
43000 rem *** feier *** ---------------------------------------------------------
43001 :
43002 :
43010 gosub10300:printspc(82)"██Jetzt beherrschen Sie alle Griffe!"
43020 printspc(2)"──────────────────────────────"
43030 poke198,0:wait198,1:gosub10300
43040 print"████┌─██▐██▐└─██s darf gefeiert werden, "nam$" !!"
43050 fora=0to1000:next:gosub44000
43060 print"██████████████████████████"
43070 fora=0to7:pokev+39+a,a+1
43080 pokev+a*2,rnd(1)*190+40:pokev+a*2+1,rnd(1)*165+70:next:pokev+21,255
43090 forb=7to0step-1:fore=0tob-3:ford=0to1:fora=0to7
43100 pokev+a*2+1,((peek(v+a*2+1)+rnd(1)*80)-40)and255
43110 pokev+a*2,((peek(v+a*2)+rnd(1)*80)-40)and255
43120 next:next
43130 pokev+23,rnd(1)*255:pokev+29,rnd(1)*255:next
43140 pokev+21,2↑(b+1)-1:next
43150 pokev+21,0:print"█":gosub10100:return
43998 :
43999 :
44000 rem *** tastatur wegschieben *** -----------------------------------------
44001 :
44002 :
44010 fora=0to40:print"█"
44020 forb=0to6:print"█ █"chr$(20):nextb:nexta
44030 return
49998 :
49999 :
50000 rem *** ende *** ----------------------------------------------------------
50001 :
50002 :
50010 gosub44000:print"█"
50020 end

ready.
```

Donkey Kong

Wie bei dem Spielhallen-Hit gleichen Namens muß auch beim TI-99/4A in der Grundversion die oberste Stufe eines Gerüstes erreicht werden. Entgegenrollenden Tonnen gilt es geschickt auszuweichen



Auf dem Weg zum Ziel stehen jeweils zwei Leitern zur Verfügung, die abwechselnd ein- und ausgeklappt werden. Nach links, rechts und oben gelangt man durch Drücken der Pfeiltasten. Sprünge sind nur im Laufen und mit der Taste "." möglich. Es gilt, ohne Zwischenfall die oberste Stufe zu erreichen.

Eine Kraftpille in der zweiten Etage verschafft eine Zeitlang Immunität gegen die gefährlichen Tonnen.

Stufe für Stufe gibt es jeweils einen Punkt. Fünfzig Punkte werden durch einen Freiläufer belohnt, die nächsten Freiläufer werden nach jeweils hundert Punkten gewährt. *Christian Schlak*

Scan: S. Höltgen


```
100 CALL CHAR(119,"1830787C7C783018")
110 FOR I=1 TO 24
120 CALL HCHAR(I,1,32,32)
130 NEXT I
140 GOTO 510
150 AA$="D O N K E y   K O N G"
160 FREE=50
170 COLOR=3
180 SZ=16-LEN(AA$)/2
190 CALL CLEAR
200 R=5
210 GOSUB 3550
220 AA$="======================"
230 SZ=16-LEN(AA$)/2
240 R=6
250 GOSUB 3550
260 AA$="PRESS ANy KEy TO BEGIN"
270 R=12
280 SZ=16-LEN(AA$)/2
290 GOSUB 3550
300 AA$="COPyRIGHT;1984;By"
310 R=19
320 SZ=16-LEN(AA$)/2
330 GOSUB 3550
340 AA$="SCHLAK & KLIETSCH"
350 R=21
360 SZ=16-LEN(AA$)/2
370 GOSUB 3550
380 CALL KEY(0,KEY,STATUS)
390 IF STATUS=0 THEN 380
400 R=22
410 FOR II=1 TO 3
420 CALL SOUND(200,330,0)
430 CALL SOUND(200,349,0)
440 CALL SOUND(200,392,0)
450 CALL SOUND(200,392,0)
460 NEXT II
470 CALL SOUND(200,523,0)
480 CALL SOUND(200,587,0)
490 CALL SOUND(400,659,0)
500 RETURN
510 REM DONKEY KONG
520 CALL CHAR(121,"00444428101010")
530 CALL COLOR(12,16,1)
540 CALL CHAR(112,"3C44524C3C1824C3")
550 CALL CHAR(113,"3C4292623C1824C3")
560 CALL CHAR(33,"FFFF4224FF2442FF")
570 CALL COLOR(1,9,1)
580 CALL SCREEN(2)
590 CALL COLOR(11,12,1)
600 CALL COLOR(9,12,1)
610 CALL COLOR(10,12,1)
620 CALL CHAR(118,"42424242424242FF")
630 CALL CHAR(40,"0000000000F132D")
640 CALL CHAR(41,"0000000000F0C8B2")
650 CALL CHAR(42,"00000101030F1F3F")
660 CALL CHAR(43,"6CA131C0E7EAE7F0")
670 CALL CHAR(44,"36858C03E757E70F")
680 CALL CHAR(45,"00008080C0F0F8FC")
690 CALL CHAR(46,"3B3B3D1F0F07071F")
700 CALL CHAR(47,"FFFFFF78F8E0FCFE")
710 CALL CHAR(88,"FFFFFF1E1F073F7F")
720 CALL CHAR(89,"DCDCBC78F0E0E0F8")
730 CALL CHAR(90,"1F1F1F0F0F7F80FF")
740 CALL CHAR(91,"FFFFFFFFE0C04080")
750 CALL CHAR(92,"FFFFFFFF07030201")
760 CALL CHAR(93,"F8F8F8F0F0FE01FF")
770 CALL COLOR(5,16,1)
780 CALL COLOR(6,16,1)
790 CALL COLOR(7,16,1)
800 MAN=3
810 POI=0
820 CALL COLOR(3,16,1)
830 CALL COLOR(4,16,1)
840 CHR=118
850 CH=1
860 CALL CHAR(117,"")
870 FOR I=96 TO 108
880 CALL CHAR(I,"")
890 NEXT I
900 A$(0)="183C427E7E423C18"
910 A$(2)="00003C5ADBDB5A3C"
920 SCHW=109
930 REI=22
940 SP=8
950 OB1=106
960 GOSUB 150
970 R=3
980 PRINT "";
990 CALL CLEAR
1000 CALL COLOR(2,COLOR,1)
1010 CALL COLOR(8,COLOR,1)
1020 CALL HCHAR(1,6,41)
1030 CALL HCHAR(2,4,42)
1040 CALL HCHAR(2,5,43)
1050 CALL HCHAR(2,6,44)
1060 CALL HCHAR(2,7,45)
1070 CALL HCHAR(3,4,46)
1080 CALL HCHAR(3,5,47)
1090 CALL HCHAR(3,6,88)
1100 CALL HCHAR(3,7,89)
1110 CALL HCHAR(4,4,90)
1120 CALL HCHAR(4,5,91)
1130 CALL HCHAR(4,6,92)
1140 CALL HCHAR(4,7,93)
1150 CALL HCHAR(1,5,40)
1160 CALL HCHAR(6,4,112,MAN)
1170 FOR I=27 TO 7 STEP -4
1180 CALL HCHAR(R,I,33,4)
1190 R=R+1
1200 NEXT I
1210 R=10
1220 FOR I=5 TO 25 STEP 4
1230 CALL HCHAR(R,I,33,4)
1240 R=R+1
1250 NEXT I
1260 R=18
1270 FOR I=27 TO 7 STEP -4
1280 CALL HCHAR(R,I,33,4)
1290 R=R+1
1300 NEXT I
1310 CTRL=27
1320 ASCII=97
1330 R=2
1340 S=30
1350 CALL HCHAR(R,S,ASCII)
1360 S=S-1
1370 ASCII=ASCII+1
1380 IF ASCII<>SCHW THEN 1400
1390 ASCII=97
1400 IF S<>CTRL-1 THEN 1350
1410 CTRL=CTRL-4
1420 R=R+1
1430 IF S<=7 THEN 1450
1440 GOTO 1350
1450 CTRL=8
1460 ASCII=97
1470 R=9
1480 S=6
1490 CALL HCHAR(R,S,ASCII)
1500 S=S+1
1510 ASCII=ASCII+1
1520 IF ASCII<>SCHW THEN 1540
1530 ASCII=97
1540 IF S<>CTRL+1 THEN 1490
1550 CTRL=CTRL+4
1560 R=R+1
1570 IF S>=28 THEN 1590
1580 GOTO 1490
1590 CTRL=27
1600 ASCII=97
1610 R=17
1620 S=28
1630 CALL HCHAR(R,S,ASCII)
1640 S=S-1
1650 ASCII=ASCII+1
1660 IF ASCII<>SCHW THEN 1680
1670 ASCII=97
1680 IF S<>CTRL-1 THEN 1630
1690 CTRL=CTRL-4
1700 R=R+1
1710 IF S<=7 THEN 1730
1720 GOTO 1630
1730 T=97
1740 IN=1
1750 AA$="DONKEy KONG:SCORE:"
1760 SZ=10
1770 R=1
1780 GOSUB 3550
1790 AA$=STR$(POI)
1800 SZ=SZ+II
1810 GOSUB 3550
1820 R=22
1830 CALL HCHAR(11,19,119)
1840 HAMMI=1000
1850 CALL CHAR(T,"")
1860 IF POI<FREE THEN 1910
1870 MAN=MAN+1
1880 CALL HCHAR(6,4,112,MAN)
1890 CALL SOUND(30,2000,0)
1900 FREE=FREE+100
1910 HAMMI=HAMMI-1
1920 IF HAMMI>0 THEN 1960
1930 HAMMI=1000
1940 CALL COLOR(11,12,1)
1950 CALL SOUND(500,200,2)
1960 CALL SOUND(40,110,7)
1970 IF REI=2 THEN 3150
1980 T=T+1
1990 CALL CHAR(T,A$(IN+1))
2000 IN=IN*-1
2010 IF T<=SCHW THEN 2030
2020 T=97
2030 CALL KEY(0,K,ST)
2040 CALL KEY(2,KJ,SSS)
2050 GOSUB 3360
2060 IF OB=T THEN 2850
2070 IF SSS<>0 THEN 2590
2080 IF ST<>0 THEN 2120
2090 FOR I=1 TO 50
2100 NEXT I
2110 GOTO 1850
2120 IF K<>68 THEN 2360
2130 IF REI<=7 THEN 2260
2140 IF REI>17 THEN 2260
2150 CALL GCHAR(REI,SP+1,OB)
2160 IF OB=T-1 THEN 2850
2170 CALL HCHAR(REI,SP,OB1)
2180 SP=SP+1
2190 IF OB<>32 THEN 2230
2200 REI=REI+1
2210 CALL GCHAR(REI,SP,OB)
2220 IF OB=T-1 THEN 2850
2230 CALL HCHAR(REI,SP,112)
2240 OB1=OB
2250 GOTO 1850
2260 CALL GCHAR(REI,SP+1,OB)
2270 IF OB=T-1 THEN 2850
2280 CALL HCHAR(REI,SP,OB1)
2290 SP=SP+1
2300 IF OB<>33 THEN 2330
2310 REI=REI-1
2320 CALL GCHAR(REI,SP,OB)
2330 CALL HCHAR(REI,SP,112)
2340 OB1=OB
2350 GOTO 1850
2360 IF K<>83 THEN 2590
2370 IF REI>=17 THEN 2490
2380 IF REI<9 THEN 2490
2390 CALL GCHAR(REI,SP-1,OB)
2400 IF OB=T THEN 2850
2410 CALL HCHAR(REI,SP,OB1)
2420 SP=SP-1
2430 IF OB<>33 THEN 2460
2440 REI=REI-1
2450 CALL GCHAR(REI,SP,OB)
2460 CALL HCHAR(REI,SP,113)
2470 OB1=OB
2480 GOTO 1850
2490 REM
2500 CALL HCHAR(REI,SP,OB1)
2510 SP=SP-1
2520 CALL GCHAR(REI,SP,OB)
2530 IF OB<>32 THEN 2560
2540 REI=REI+1
2550 CALL GCHAR(REI,SP,OB)
2560 CALL HCHAR(REI,SP,113)
2570 OB1=OB
2580 GOTO 1850
2590 IF KJ<>13 THEN 1850
2600 IF K=68 THEN 2620
2610 IF K<>83 THEN 1850
2620 CALL HCHAR(REI,SP,OB1)
2630 CALL SOUND(100,-4,1)
2640 REI=REI-1
2650 CALL GCHAR(REI,SP,NICO)
2660 IF NICO<>119 THEN 2690
2670 HAMMI=10
2680 CALL COLOR(11,14,1)
2690 CALL HCHAR(REI,SP,112)
2700 CALL HCHAR(REI,SP,32)
2710 IF REI>15 THEN 2770
2720 IF REI<9 THEN 2770
2730 SP=SP-2
2740 CALL GCHAR(REI+1,SP,GH)
2750 IF GH=33 THEN 2810 ELSE 2800
2760 GOTO 2810
2770 SP=SP+2
2780 CALL GCHAR(REI+1,SP,GH)
2790 IF GH=33 THEN 2810
2800 REI=REI+1
2810 CALL GCHAR(REI,SP,OB)
2820 CALL HCHAR(REI,SP,112)
2830 OB1=OB
2840 GOTO 1850
2850 IF HAMMI>10 THEN 2870
2860 GOTO 1850
2870 MAN=MAN-1
2880 SP=7
2890 CALL CHAR(T,"")
2900 T=97
2910 CALL SOUND(500,500,0,1000,3,110,7)
2920 POI=POI+(24-REI)
2930 REI=22
2940 IF MAN>0 THEN 970
2950 AA$="GAME OVER"
2960 CALL HCHAR(6,2,32,4)
2970 R=12
2980 SZ=16-(LEN(AA$)/2)
2990 GOSUB 3550
3000 AA$="POINTS:"
3010 R=10
3020 SZ=16-LEN(AA$)/2
3030 GOSUB 3550
3040 AA$=STR$(POI)
3050 R=10
3060 SZ=SZ+8
3070 GOSUB 3550
3080 PRINT " PRESS ANy KEy TO GO BACK";
3090 FOR II=1 TO 1000
3100 CALL KEY(0,KEY,STATUS)
3110 IF STATUS<>0 THEN 110
3120 NEXT II
3130 PRINT
3140 GOTO 110
3150 AA$="GESCHAFFT!"
3160 IF COLOR=16 THEN 3180
3170 COLOR=COLOR+1
3180 R=12
3190 SZ=16-LEN(AA$)/2
3200 GOSUB 3550
3210 POI=POI+(24-REI)
3220 AA$=STR$(POI)
3230 R=10
3240 SZ=16
3250 GOSUB 3550
3260 AA$="PUNKTE"
3270 SCHW=SCHW-1
3280 SZ=16-LEN(AA$)/2
3290 R=8
3300 GOSUB 3550
3310 REI=22
3320 SP=7
3330 FOR CHRI=1 TO 200
3340 NEXT CHRI
3350 GOTO 970
3360 COUNT=COUNT+1
3370 IF COUNT<5 THEN 3450
3380 COUNT=0
3390 CH=CH*-1
3400 CHR=CHR+CH
3410 REM
3420 CALL VCHAR(7,12,CHR,3)
3430 CALL VCHAR(14,23,CHR,4)
3440 CALL HCHAR(REI,SP,112)
3450 IF K=69 THEN 3470
3460 RETURN
3470 CALL GCHAR(REI-1,SP,NI)
3480 IF NI=118 THEN 3500
3490 RETURN
3500 CALL HCHAR(REI,SP,OB1)
3510 OB=0
3520 REI=REI-1
3530 CALL HCHAR(REI,SP,112)
3540 RETURN
3550 FOR II=1 TO LEN(AA$)
3560 CALL HCHAR(R,II+SZ,ASC(SEG$(AA$,II,
1)))
3570 NEXT II
3580 RETURN
```

Scan: S. Höltgen


```
1076 IFZ$=CHR$(133)THENGOSUB800:GOTO1075
1077 IFZ$=CHR$(134)THENGOSUB850:GOTO1076
1078 IFZ$=CHR$(135)THENIFSA<>1THENONVAL(X$)GOSUB1084,1085,1086:GOTO1000
1079 IFZ$=CHR$(135)THENGOSUB1084:RETURN
1080 IFZ$=CHR$(136)THEN615
1081 IFZ$<>CHR$(20)THEN1087
1082 IFSA=1THENGOSUB1084:GOTO1000
1083 ONVAL(X$)GOSUB1084,1085,1086:GOTO1040
1084 POKE1024+40*(T+X)+29,32:POKE1024+40*(T+X)+30,32:RETURN
1085 POKE1024+40*(T+X)+33,32:POKE1024+40*(T+X)+34,32:RETURN
1086 POKE1024+40*(T+X)+37,32:POKE1024+40*(T+X)+38,32:RETURN
1087 Y=VAL(X$):TA$(B,X,Y)=RIGHT$(STR$(ER),2):RETURN
1100 ONXGOTO1110,1120,1130,1140,1150,1160,1170,1180,1190,1200,1210,1220,1230
1110 ER=0:FORY=1TO5:IFW(Y)=1THENER=ER+1
1115 NEXT:RETURN
1120 ER=0:FORY=1TO5:IFW(Y)=2THENER=ER+2
1125 NEXT:RETURN
1130 ER=0:FORY=1TO5:IFW(Y)=3THENER=ER+3
1135 NEXT:RETURN
1140 ER=0:FORY=1TO5:IFW(Y)=4THENER=ER+4
1145 NEXT:RETURN
1150 ER=0:FORY=1TO5:IFW(Y)=5THENER=ER+5
1155 NEXT:RETURN
1160 ER=0:FORY=1TO5:IFW(Y)=6THENER=ER+6
1165 NEXT:RETURN
1170 FORY=1TO6:ER=0:FORZ=1TO5:IFW(Z)=YTHENER=ER+1
1174 NEXT:IFER>2THENER=0:FORY=1TO5:ER=ER+W(Y):NEXT:RETURN
1176 ER=0:NEXT:ER=0:RETURN
1180 FORY=1TO6:ER=0:FORZ=1TO5:IFW(Z)=YTHENER=ER+1
1184 NEXT:IFER>3THENER=0:FORY=1TO5:ER=ER+W(Y):NEXT:RETURN
1186 NEXT:ER=0:RETURN
1190 E=0:T=0
1191 FORY=1TO6:IFY=TTHEN1199
1192 FORZ=1TO5:IFW(Z)=YTHENE=E+1
1193 NEXT:IFT<>0THEN1196
1194 IFE<>2ANDE<>3THEN1197
1195 T=Y:ER=E:E=0:GOTO1191
1196 IFER+E=5THENER=25:RETURN
1197 E=0
1199 NEXTY:ER=0:RETURN
1200 FORE=1TO3:ER=0:FORY=ETOE+3:FORZ=1TO5:IFW(Z)=ER+ETHENER=ER+1
1204 NEXT:IFER>3THENER=30:RETURN
1206 NEXTY,E:ER=0:RETURN
1210 FORE=1TO2:ER=0:FORY=ETOE+4:FORZ=1TO5:IFW(Z)=ER+ETHENER=ER+1
1214 NEXT:IFER=5THENER=40:RETURN
1216 NEXTY,E:ER=0:RETURN
1220 FORY=2TO5:IFW(Y)=W(1)THENNEXT:ER=50:RETURN
1225 ER=0:RETURN
1230 ER=W(1)+W(2)+W(3)+W(4)+W(5):RETURN
2000 PRINT"]MAN KANN MIT 3 VERSUCHEN 13 MOEGLICHKEI-TEN WUERFELN  UND IN DIE "
2010 PRINT"TABELLE EINTRA-GEN. MAN KANN BIS ZU 3 SPALTEN BENUTZEN.";
2020 PRINT"2. SPALTE = DOPPELT;  3. = DREIFACH. GE-WONNEN HAT DERJENIGE,  ";
2030 PRINT"DER DIE  HOECHSTEPUNKTZAHL ERREICHT HAT. ";
2040 PRINT"TABELLE:"
2050 PRINT" 1 EINSER: ALLE 1ER ADDIEREN"
2060 PRINT" 2 ZWEIER: ALLE 2ER ADDIEREN"
2070 PRINT" 3 DREIER: ALLE 3ER ADDIEREN"
2080 PRINT" 4 VIERER: ALLE 4ER ADDIEREN"
2090 PRINT" 5 FUENFER: ALLE 5ER ADDIEREN"
2100 PRINT" 6 SECHSER: ALLE 6ER ADDIEREN"
2110 PRINT" 7 DREIERPASCH: ALLE WUERFEL ADDIEREN      (WENN MINDESTENS 3 ";
2120 PRINT"GLEICHE VORHANDEN)";
2130 PRINT" 8 VIERERPASCH: ALLE WUERFEL ADDIEREN      (WENN MINDESTENS 4 ";
2140 PRINT"GLEICHE VORHANDEN)";
2150 PRINT" 9 FULL HOUSE: 2  GLEICHE WUERFEL  UND 3   GLEICHE WUERFEL ";
2160 PRINT"(25 PUNKTE)"
2170 PRINT"10 KLEINE STRASSE: 4ER FOLGE; 1234, 2345   ODER 3456 (30 PUNKTE)"
2180 PRINT"11 GROSSE STRASSE: 5ER FOLGE; 12345 ODER   23456 (40 PUNKTE)"
2190 PRINT"12 YAHTZEE: ALLE WUERFEL GLEICHE  AUGEN-   ZAHL (50 PUNKTE)"
2200 PRINT"13 CHANCE: ALLE WUERFEL ADDIEREN";
2210 GETX$:IFX$<>""THENRETURN
2220 GOTO2210

READY.
```

Planetensicherung

Mit Kampfgeschützen sollen Feinde vom Planeten Krona abgewehrt werden. Eine Aufgabe, die nicht einfach zu bewältigen ist. Das Spiel läuft auf der Grundversion des TI-99/4A

Nach RUN erscheint am oberen Bildrand Ihr Mutterschiff mit dem angekoppelten Kampfgeschütz. Sie müssen nun versuchen, das Kampfgeschütz sicher auf den am unteren Bildrand zu sehenden Planeten zu bringen.

Schwierige weiche Landung

Mit der Taste „A" koppeln Sie das Kampfgeschütz ab, und mit der Taste „B" können Sie bremsen, womit Sie eine weiche Landung garantieren können. Aber Vorsicht: Eine nur etwas zu starke Bremsung reicht aus, um das Kampfgeschütz hilflos im Raum stehen zu lassen. Hier heißt es deshalb, das gute Mittelmaß zu behalten, denn bei zu schwacher Bremsung zerschellt das Kampfgeschütz an der Planetenoberfläche.

Nach einer geglückten Landung ertönt eine Fanfare, und dann rückt auch schon der Feind an, der Sie mit Laserstrahlen beschießt. Setzen Sie sich zur Wehr! Weichen Sie den Laserstrahlen mit den Tasten „S" und „D" in Pfeilrichtung aus, und schalten Sie Ihren Angreifer mit Ihrem Laser aus, den Sie mit der Punkttaste aktivieren.

Zusätzliches Geschütz

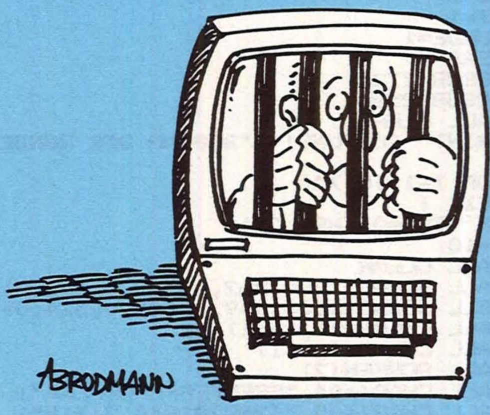
Aber passen Sie auf, daß der Feind nicht landet, denn dann ist das Spiel aus, und der Computer teilt die erreichte Punktzahl mit. Sobald Sie 7 Feinde abgeschossen haben, beginnt die nächste Runde, die noch schwieriger ist, und Sie bekommen ein zusätzliches Kampfgeschütz.

Werden Sie vom Feind abgeschossen, müssen Sie wieder von vorn anfangen. Nach dem Ende des Spiels können Sie beliebig oft Ihre Mission wiederholen und so das Punktekonto verbessern. *M. Strunk*

```
100 REM  PLANETENSICHERUNG
110 WW=0
120 ST=1
130 RUNDE=1
140 VERS=3
150 GOSUB 2720
160 GOSUB 2580
170 CALL CLEAR
180 PRINT "ACHTUNG!":"BEGINN DER RUNDE";
RUNDE
190 FOR I=1 TO 700
200 NEXT I
210 IF VERS=0 THEN 2410
220 D=10
230 CALL CLEAR
240 CALL SOUND(1000,262,0,330,0,392,0)
250 CALL SOUND(1000,349,0,440,0,524,0)
260 CALL COLOR(10,16,1)
270 CALL COLOR(9,7,1)
280 CALL SCREEN(2)
290 CALL CHAR(104,"FFFFFFFFFFFFFFFF")
300 CALL CHAR(105,"1248281844121318")
310 CALL CHAR(106,"C3C3FFE7E7FFFF3C")
320 CALL CHAR(107,"181866FF99FF1818")
330 CALL CHAR(103,"1818181818181818")
340 CALL CHAR(108,"183C1818FFFF2466")
350 CALL HCHAR(3,2,108,VERS-1)
360 CALL HCHAR(24,1,104,32)
370 FOR I=1 TO 30
380 CALL HCHAR(1,I,106,3)
390 CALL HCHAR(2,I+1,108)
400 CALL KEY(0,K,S)
410 IF S=0 THEN 430
420 IF K=65 THEN 470
430 CALL HCHAR(1,I,32,3)
440 CALL HCHAR(2,I+1,32)
450 NEXT I
460 GOTO 370
470 FOR A=2 TO 23
480 CALL HCHAR(A,I+1,108)
490 FOR R=1 TO D
500 NEXT R
510 CALL HCHAR(A,I+1,32)
520 CALL KEY(0,K,S)
530 IF S=0 THEN 570
540 IF K=66 THEN 550 ELSE 570
550 D=D+20
560 IF D=170 THEN 680
570 NEXT A
580 IF D<120 THEN 600
590 GOTO 760
600 CALL HCHAR(23,I+1,105)
610 FOR W=1 TO 20
620 CALL SOUND(-100,-7,W)
630 NEXT W
640 CALL SCREEN(7)
650 PRINT "ZU HART GELANDET"
660 VERS=VERS-1
670 GOTO 190
680 CALL HCHAR(A,I+1,108)
690 FOR W=1 TO 20
700 CALL SOUND(-100,-6,W)
710 NEXT W
720 CALL SCREEN(15)
730 VERS=VERS-1
740 PRINT "ZU STARK GEBREMST"
750 GOTO 190
760 CALL SCREEN(15)
770 CALL HCHAR(23,I+1,108)
780 CALL SOUND(1000,262,0,330,0,392,0)
790 PRINT "GUT GELANDET":"BONUS:300"::
800 WW=WW+300
810 FOR U=1 TO 2
820 CALL SOUND(800,262,0,330,0,392,0)
830 CALL SOUND(400,349,0,440,0,524,0)
840 CALL SOUND(400,392,0,494,0,294,0)
850 NEXT U
860 CALL CLEAR
870 CALL SCREEN(2)
880 CALL HCHAR(3,2,108,VERS-1)
890 CALL HCHAR(24,1,104,32)
900 V=15
910 Y=16
920 Z=5
930 IF Z>23 THEN 2410
940 IF Z<2 THEN 1100
950 IF Y>32 THEN 1160
960 IF Y<2 THEN 1140
970 RANDOMIZE
980 CALL HCHAR(Z,Y,107)
990 CALL HCHAR(23,V,108)
1000 X=INT(RND*6)+1
1010 D=INT(RND*E)+1
1020 CALL KEY(0,K,S)
1030 IF S=0 THEN 1070
1040 CALL HCHAR(23,V,32)
1050 V=V-(K=68)+(K=83)
1060 IF K=46 THEN 2060
1070 IF D=1 THEN 1180
1080 CALL HCHAR(Z,Y,32)
1090 ON X GOTO 1100,1100,1100,1120,1140,
1160
1100 Z=Z+ST
1110 GOTO 930
1120 Z=Z-ST
1130 GOTO 930
1140 Y=Y+ST
1150 GOTO 930
1160 Y=Y-ST
1170 GOTO 930
1180 CALL SOUND(100,-6,0)
1190 CALL VCHAR(Z,Y,103,(24-Z))
2000 CALL HCHAR(23,V,108)
2010 IF Y=V THEN 2130
2020 CALL HCHAR(Z,Y,107)
2030 CALL SOUND(50,-3,0)
2040 CALL VCHAR(Z,Y,32,(24-Z))
2050 GOTO 1080
2060 CALL SOUND(100,-5,0)
2070 CALL VCHAR(1,V,103,23)
2080 IF V=Y THEN 2230
2090 CALL HCHAR(23,V,108)
2100 CALL SOUND(50,-1,0)
2110 CALL VCHAR(1,V,32,23)
2120 GOTO 1070
2130 CALL SOUND(1000,-6,0)
2140 CALL HCHAR(23,V,105)
2150 VERS=VERS-1
2160 IF VERS=0 THEN 2170
2170 CALL VCHAR(Z,Y,32,(24-Z))
2180 FOR I=1 TO 16
2190 CALL COLOR(10,I,1)
2200 CALL SOUND(200,110,5,120,3,-5,0)
2210 NEXT I
2220 GOTO 190
2230 CALL HCHAR(Z,Y,105)
2240 IN=IN+1
2250 IF IN=7 THEN 2300
2260 CALL SOUND(100,110,5,120,3,-6,0)
2270 WW=WW+50
2280 CALL VCHAR(1,V,32,23)
2290 GOTO 810
2300 ST=ST+1
2310 RUNDE=RUNDE+1
2320 IN=0
2325 WW=WW+1000
2330 VERS=VERS+1
2340 CALL SCREEN(16)
2350 PRINT "BRAVO!":"BONUS:1000":"1 SOND
ERGESCHUETZ":"SCORE:";WW
2360 PRINT "ACHTUNG!":"BEGINN DER RUNDE"
;RUNDE
2370 FOR I=1 TO 2000
2380 NEXT I
2390 CALL SOUND(1000,262,0,330,0,392,0)
2400 GOTO 810
2410 CALL CLEAR
2420 CALL SCREEN(8)
2430 PRINT "SIE HABEN IHREN AUFTRAG"
2440 IF WW>1000 THEN 2470
2450 PRINT "SCHLECHT"
2460 GOTO 2510
2470 IF WW>3000 THEN 2500
2480 PRINT "GUT"
2490 GOTO 2510
2500 PRINT "HERVORRAGENT"
2510 PRINT "AUSGEFUEHRT"
2520 PRINT "SCORE:";WW
2530 PRINT "NOCH EIN SPIEL ?":"DRUECKEN
SIE J ODER N"
2540 CALL KEY(0,K,S)
2550 IF S=0 THEN 2540
```

Scan: S. Höltgen


```
2560 IF K=74 THEN 100
2570 END
2580 CALL CLEAR
2590 PRINT "WELCHEN SCHWIERIGKEITSGRAD?"
:"LEICHT(L)":"MITTEL(M)":"SCHWER(S)"
2600 CALL KEY(0,K,S)
2610 IF S=0 THEN 2600
2620 IF K<>76 THEN 2650
2630 E=6
2640 RETURN
2650 IF K<>77 THEN 2680
2660 E=3
2670 RETURN
2680 IF K<>83 THEN 2710
2690 E=1
2700 RETURN
2710 GOTO 2600
2720 CALL SCREEN(15)
2730 CALL CLEAR
2740 PRINT "AUFTRAG PLANETENSICHERUNG"::
:
2750 PRINT "PUNKTETAFEL":"LANDUNG      =
 300 P.":"FEIND         =  50 P."
2760 PRINT "RUNDENWECHSEL=1000 P.":::"VI
EL ERFOLG!":"TASTE DRUECKEN!":::
2770 CALL KEY(0,K,S)
2780 IF S=0 THEN 2770
2790 RETURN
```



Type-Writer

Ein amüsantes Reaktionsspiel für den TI-99/4A mit Extended-BASIC: Ein Zeichen saust den Bildschirm hinunter und kann nur durch einen entsprechenden Tastendruck vor dem Verschwinden in einer bereitstehenden Kiste bewahrt werden

Jeder richtige Tastendruck wird mit einer Punktzahl zwischen 10 und 99 honoriert. Bei mehr als 2000 Zeichen ist das Spiel beendet und das Ergebnis angezeigt.

Hat man gelernt, auch bei höheren Geschwindigkeiten der Zeichen richtig zu reagieren, dann stellt die Tastatur des TI-99/4A kein Problem mehr dar. *Andreas Molisch*

```
100 !-----------------------
110 !TYPE  WRITER
120 !************************
130 !CO. BY A.MOLISCH
140 !-----------------------
150 DIM A$(10,27),G(27),C$(62)
160 CALL CLEAR :: CALL SCREEN(2):: FOR I
=0 TO 13 :: CALL COLOR(I,16,2):: NEXT I
:: CALL COLOR(14,9,2):: CALL MAGNIFY(2)
170 CALL CHAR(136,RPT$("F",16))
180 !----------------------
190 !TITEL
200 !----------------------
210 D=3 :: C=0
220 CALL CLEAR :: CALL DELSPRITE(ALL)
230 DISPLAY AT(8,10):"TYPE  WRITER" :: F
OR I=1 TO 10 :: CALL SOUND(-50,110*I,0):
: NEXT I :: FOR I=10 TO 1 STEP -1 :: CAL
L SOUND(-50,110*I,0):: NEXT I
240 DISPLAY AT(10,9):"BITTE WAEHLEN:"
250 DISPLAY AT(12,10):"[1] ZUFALL"
260 DISPLAY AT(14,10):"[2] BEFEHLE"
270 DISPLAY AT(16,10):"[3] EIGENE WOERTE
R"
```

```
280 ACCEPT AT(10,23)VALIDATE(DIGIT)BEEP
SIZE(1):A :: IF A<1 OR A>3 THEN 280
290 CALL CLEAR
300 DISPLAY AT(9,9):"GESCHWINDIGKEIT?"
310 DISPLAY AT(11,12):"(10-99)"
320 ACCEPT AT(11,20)VALIDATE(DIGIT)BEEP
SIZE(2):B :: IF B<10 THEN 320
330 CALL CLEAR
340 !----------------------
350 !BILDSCHIRMAUFBAU
360 !----------------------
370 CALL DELSPRITE(ALL)
380 DISPLAY AT(4,3):"PUNKTE:"
390 DISPLAY AT(6,4):USING "#####":C
400 DISPLAY AT(9,3):"GESCHW.:"
410 DISPLAY AT(11,5):B
420 DISPLAY AT(14,4):CHR$(136);":";D
430 CALL SPRITE(#28,136,9,176,150)
440 DISPLAY AT(14,15):"TASTE DRUECKEN" :
: CALL KEY(0,K,S):: IF S=0 THEN 440
450 DISPLAY AT(14,15):" "
460 !----------------------
470 !AUSWAHL
480 !----------------------
490 ON A GOTO 500,580,750
500 !----------------------
510 !ZUFALL
520 !----------------------
530 RANDOMIZE :: FOR I=1 TO 10
540 G(I)=INT((90-48+1)*RND)+48
550 IF G(I)=58 OR G(I)=59 OR G(I)=60 OR
G(I)=61 OR G(I)=62 OR G(I)=63 OR G(I)=64
 THEN 540
560 NEXT I :: CALL SOUND(1000,110,0)
570 M=1 :: GOTO 1110
580 !----------------------
590 !BEFEHLE
```

Scan: S. Höltgen


```
600 !----------------------
610 DATA ACCEPTAT,CALLCHAR,CALLCHARPAT,C
ALLCHARSET,CALLCLEAR,CLOSE,CALLCOINC,CAL
LCOLOR,DATA,DEF,CALLDELSPRITE,DIM,DISPLA
YAT,DISPLAYUSING
620 DATA CALLDISTANCE,END,CALLERR,FORI=1
TO10,CALLGCHAR,GOSUB,GOTO,CALLHCHAR,IFTH
ENELSE,IMAGE,CALLINIT,INPUT,CALLJOYST
630 DATA CALLKEY,CALLLINK,LINPUT,CALLLOA
D,CALLLOCATE,CALLMAGNIFY,CALLMOTION,NEXT
I,ONBREAK,ONERROR,ONAGOSUB,ONAGOTO,ONWAR
NING
640 DATA OPEN,OPTIONBASE,CALLPATTERN,CAL
LPEEK,CALLPOSITION,PRINT,PRINTUSING,RAND
OMIZE,READ,REM,RESTORE
650 DATA RETURN,CALLSCREEN,CALLSOUND,CAL
LSPRITE,STOP,SUB,SUBEND,SUBEXIT,CALLVCHA
R,CALLVERSION
660 !----------------------
670 !BEFEHLEEINLESEN
680 !----------------------
690 RESTORE :: FOR I=1 TO 61 :: READ C$(
I):: NEXT I :: CALL SOUND(1000,110,0)
700 RANDOMIZE :: T=INT(61*RND+1):: FOR I
=1 TO LEN(C$(T)):: G(I)=ASC(SEG$(C$(T),I
,1)):: NEXT I
710 M=2 :: GOTO 1110
720 !----------------------
730 !EIGENE WOERTER
740 !----------------------
750 CALL DELSPRITE(#28):: DISPLAY AT(17,
10):"NEUE EINGABEN?(J/N)"
760 CALL KEY(0,K,S):: IF K=78 AND E$(1)=
"" THEN 760 ELSE IF K=78 THEN 810 ELSE I
F K<>74 THEN 760
770 DISPLAY AT(17,10):"WIEVIELE EINGABEN
?"
780 DISPLAY AT(19,15):"(1-9)"
790 ACCEPT AT(19,21)VALIDATE(DIGIT)SIZE(
1)BEEP:F :: IF F<1 THEN 790
800 FOR I=1 TO F :: DISPLAY AT(17,10):"W
ORT";I :: ACCEPT AT(19,3)SIZE(27)BEEP:E$
(I):: NEXT I :: DISPLAY AT(19,3):" "
810 DISPLAY AT(17,10):"TASTE DRUECKEN" :
: CALL KEY(0,K,S):: IF S THEN DISPLAY AT
(17,10):" " ELSE 810
820 !----------------------
830 !WORTAUSWAHL
840 !----------------------
850 CALL SPRITE(#28,136,9,176,150)
860 RANDOMIZE :: J=INT(F*RND+1):: FOR I=
1 TO LEN(E$(J)):: G(I)=ASC(SEG$(E$(J),I,
1)):: NEXT I :: CALL SOUND(1000,110,0)
870 IF E$(J)="" THEN 800 :: M=3 :: GOTO
1110
880 !----------------------
890 !FEHLER
900 !----------------------
910 FOR I=1 TO 20 STEP 4 :: CALL SOUND(-
50,110,I,120,I,115,I,-7,I):: NEXT I :: D
=D-1 :: IF D<1 THEN 1040 :: GOTO 340
920 !----------------------
930 !ALLE GESCHAFFT
940 !----------------------
950 CALL DELSPRITE(ALL):: FOR I=1 TO 10
:: CALL SOUND(-50,110*I,0):: NEXT I
960 DISPLAY AT(15,15):"HERZLICHEN"
970 DISPLAY AT(17,12):"GLUECKWUNSCH!!!"
980 DISPLAY AT(20,13):"NOCHMAL?(J/N)"
990 CALL KEY(0,K,S):: IF K=74 THEN 180 E
LSE IF K=78 THEN CALL CLEAR ELSE 990
1000 END
1010 !----------------------
1020 !NICHT ALLE GESCHAFFT
1030 !----------------------
1040 CALL DELSPRITE(ALL):: FOR I=10 TO 1
 STEP -1 :: CALL SOUND(-50,110*I,0):: NE
XT I
1050 DISPLAY AT(6,4):USING "#####":C
1060 DISPLAY AT(14,4):CHR$(136);":";D
1070 DISPLAY AT(15,15):"SCHADE!!!"
1080 DISPLAY AT(18,13):"NOCHMAL?(J/N)"
1090 CALL KEY(0,K,S):: IF K=74 THEN 180
ELSE IF K=78 THEN CALL CLEAR ELSE 1090
1100 END
1110 !----------------------
1120 !SPIEL
1130 !----------------------
1140 I=1
1150 CALL SPRITE(#I,G(I),11,1,150,B,0)
1160 CALL KEY(0,K,S):: IF K=G(I)THEN 117
0 :: CALL COINC(ALL,E):: IF E THEN 910 :
: GOTO 1160
1170 CALL DELSPRITE(#I):: C=C+B :: DISPL
AY AT(6,4):USING "#####":C :: IF C>20000
 THEN 950 :: I=I+1 :: IF M=1 AND I<=10 T
HEN 1150
1180 IF M=2 AND I<=LEN(C$(T))THEN 1150 E
LSE IF M=3 AND I<=LEN(E$(J))THEN 1150
1190 B=B+1 :: IF B>99 THEN B=99
1200 DISPLAY AT(11,5):B
1210 CALL SOUND(500,110,0):: CALL SOUND(
500,200,0)
1220 ON A GOTO 500,580,820
```

Achtung!!!

RADIX

RADIX
Bürotechnik
Handelsgesellschaft mbH
Bornstraße 4 · 2000 Hamburg 13
Tel. 040/44 16 95 · Telex 21 36 82 radix d
tägl. 10.00–12.30 + 13.30–18.30 Uhr
Sa. 10.00–13.00 Uhr

Original TI-Produkte · Fremdanbieter für TI 99/4A · Importartikel für TI 99/4A

Umfangreiches Angebot für Texas Instruments
TI 99/4A

Hardware:
TI-Box, Disklaufw., Contr.
32 K-Erw.+10 Disk. **1698,–**
32 K-Erw. extern **428,–**
Druckerinterface extern **348,–**
Drucker GP 50 + Kabel + Druckerinterface extern **798,–**
dito mit GP 550 **1198,–**
dito mit EPSON RX 80 **1448,–**
dito mit EPSON FX 80 **1990,–**
Graphik Tableau **298,–**
Sprachsteuereinheit **348,–**
RGB-Monitor+Modulator **1030,–**
Superjoyst. einfach **39,–**
zweifach **75,–**
Mini Memory **295,–**
Terminal Emulator **199,–**

Software:
Editor Assembler **189,–**
Extendet Basic lieferbar! **295,–**
Parsec **79,–**
Alpiner **79,–**
Tunnels of Doom **79,–**
Adventure Modul **79,–**
TI-Logo II **320,–**
TI-Writer **320,–**
Mathematik I **99,–**
Car Wars **39,–**
Munch Man **79,–**
Tomstone City **39,–**

US-Spiele:
Moon Mine, Sewermania, Bigfoot, Meteor Belt, M*A*S*H, Microsurgion, Demon Attack, Hopper, Star Treck, Jawbreaker II, Slymoids, Munchmobile, Moonsweeper, Baseball*, Terry Turtle's Adventure*,
Fathom Burgertime, Pirats-island, Bad Rogers
je Modul **99,–**
***Sprachsteuereinheit notwendig!**

Atari-Spiele TI 99:
Defender, Donkey Kong, DIG-DUG, Moon Patrol, Pac Man, Centipede, Mrs. Pac Man, Pool Position
Jungle Hunt
je Modul **99,–**

Sonstiges:
Flugsimulation **49,–**
Staubschutzhaube aus Kunstleder **29,–**
und, und, und, und!
Graphik Tableau für TI 99 = **298,–**
für VC 64 = **298,–**

SHARP
Pocket Computer
PC 1500 375,– CE 150,– PC 1500 A,
PC 1245, PC 1251, CE 125 PC 1401, CE 126 p
PC 1212, CE 122
Sowie Systemhandbücher + Zubehör

SHARP MZ-700
MZ 731, Personal-Computer, 64 kB-RAM inkl. Kassettenrekorder, Farbgrafik-Drucker, eingebauter HF-Modulator und RGB-Anschluß **1190,–**
SFD 700 – Floppy für MZ-700 mit 280 KB **1390,–**

Brother, Silver Reed, Seikosha, HP, 3 M Scotch, Apple, Sanyo, Commodore, Atari
Zubehör:
Monitore, Disketten, Drucker-buffer, Disketten-Organisation, Computer-Möbel, Papier

Abdeckhauben
Kunstleder mit Leinen
für TI 99/4A, CBM 64, Atari 600, MZ 700 **29,90**
FX 80 **39,90** QX 10 **99,00**
Sonderanfertigungen kein Problem!

Preise Stand 1. 6. 84 Alle Preise incl. MwSt. · Preisliste anfordern! · Lieferung erfolgt per NN oder gegen Verrechnungsscheck.
Bestellungen über DM 500,– werden frei Haus geliefert, unter DM 500,– werden DM 5,– Versandpauschale berechnet.
RADIX Bürotechnik · Bornstraße 4 + Heinrich-Barth-Straße 13 · 2000 Hamburg 13 · Telefon 040/4416 95 · Telex 21 36 82 RADIX

Scan: S. Höltgen

Sound-Effekte für Computer-Spiele

Mit nur wenigen Programmzeilen läßt sich die Wirkung eines Spieles enorm verbessern

Jeder Computer hat im Sound-Bereich andere Befehle, mit denen die Tongeneratoren eingeschaltet werden. Trotzdem gibt es einige Prinzipien, die bei allen Modellen gleich sind. Wenn zum Beispiel der Sound für ein herunterfallendes Objekt erzeugt werden soll, eignet sich am besten eine Melodienfolge, die vom hohen zum tiefen Ton verläuft. Der Ablauf muß so schnell sein, daß die einzelnen Töne nicht zu hören sind, sondern nur ein längerer „Pfiff". Um das zu erreichen, werden sämtliche Töne mit einer FOR...NEXT-Schleife aufgerufen und abgespielt.

Um ein Gefahrensignal zu erzeugen, wird das Prinzip vom „fallenden Objekt" genau umgedreht. Nicht die Tonfolge von hoch und tief, sondern umgekehrt ist nötig, um ein Alarmsignal zu erzeugen, wie Sie es sicher aus Filmen von U-Boot- oder Kraftwerkskatastrophen kennen. Wird ein solcher Alarm-Pfiff durch eine FOR...NEXT-Schleife dreimal wiederholt, hört es sich sehr „gefährlich" an.

Ganz nach Lust und Laune

Ein Fahrzeuggeräusch entsteht durch Rauschen, das durch Filtern zu Summen oder Brummen wird. Wenn unterschiedliche Töne nacheinander auf diese Weise verzerrt werden, simuliert dies eine „Gangschaltung", wobei ein tiefer Ton einem niedrigen Gang und ein hoher Ton einem hohen Gang entspricht.

Natürlich können Sie die Sound-Beispiele auch anders einsetzen. So eignet sich das Geräusch für fallendes Objekt auch für einen Zusammenstoß. Probieren Sie die Sounds einfach aus, und setzen Sie sie ganz nach Lust und Laune ein.

ZX Spectrum

```
REM PUNKTVERLUST/FEHLSCHLAG
10 FOR x=12 TO 1 STEP -1
20 BEEP 0.05,x
30 NEXT x
```

Mit x=12 TO 1 sind die Töne C bis Cis gemeint; und zwar rückwärtslaufend (STEP −1). Durch BEEP 0.05,x werden die Töne relativ schnell abgespielt. Die Zeit 0.05 können Sie nach Belieben ändern.

```
REM SCHNARRENDES GERÄUSCH
10 FOR x=20 TO -6 STEP -1
20 BEEP 0.005,x
30 NEXT x
```

Das gleiche Prinzip wie im vorhergehenden Beispiel. Allerdings ist hier der Tonumfang vergrößert (von 20 bis −6) und die Ablaufgeschwindigkeit erheblich verkürzt worden (BEEP 0.005,x). Dadurch entsteht ein schnarrendes Geräusch, das sich für verschiedene Situationen einsetzen läßt (zum Beispiel, wenn eine Figur gegen eine Labyrinthwand rennt oder mit einer anderen zusammenstößt).

```
REM GEFAHR/ALARM
10 FOR y=0 TO 2
20 FOR x=0 TO 12
30 BEEP 0.02,x
40 NEXT x
50 NEXT y
```

Mit x=0 TO 20 wird eine ganze Tonleiter aufgerufen, und zwar chromatisch (in Halbtonschritten). In Zeile 30 wird der aktuelle Ton abgespielt. Die Zeit von 0.02 können Sie nach Belieben ändern. Durch die FOR...NEXT-Schleife in Zeile 10 und 50 wird der ganze Abspielvorgang dreimal aufgerufen.

```
REM FALLENDES OBJEKT
10 FOR x=24 TO 12 STEP -0.1
20 BEEP 0.01,x
30 NEXT x
```

Hier geschieht genau das, was am Anfang in den Prinzipien erklärt wurde. Die Abspielgeschwindigkeit von 0.01 in Zeile 20 können Sie beliebig ändern.

```
PUNKTGEWINN/SIEGESMELODIE
10 FOR y=0 TO 4
20 READ d,t
30 BEEP d,t
40 NEXT y
50 DATA 0.1,7,0.1,4,0.1,4
60 DATA 0.5,9,0.6,7
```

In diesem kleinen Programm wird eine kurze Melodie abgespielt, die aus den Tönen G (= 7), E (= 4), F (= 5) und A (= 9) besteht. Die FOR....NEXT-Schleife in Zeile 10 und 40 korrespondiert mit dem READ d,t-Befehl in Zeile 20; d,t steht als Abkürzung für „Dauer, Ton". In den DATA-Zeilen stehen die Werte für Dauer und Ton, die bei jedem FOR....NEXT-Durchlauf nacheinander gelesen und gespielt werden.

```
REM NIEDERLAGE
10 FOR y=0 TO 9
20 READ d,t
30 BEEP d,t
40 NEXT y
50 DATA 0.4,9,0.4,9,0.2,9,0.4,9
60 DATA 0.4,12,0.2,11,0.4,11
70 DATA 0.4,9,0.2,9,0.4,9
```

Hier werden nach dem gleichen Prinzip wie im vorhergehenden Bei-

Scan: S. Höltgen

spiel Spielzeiten und Töne aufgerufen; jedoch ist diese Melodie ein „Trauermarsch", was gut zur Niederlage oder zum Punktverlust in einem Spiel paßt.

Atari

Da bei Atari die neuen Modelle 600 XL und 800 XL im Sound-Bereich mit den alten identisch sind, gelten die nachfolgenden Listings für alle Atari-Computer. Als besonderer Komfort stehen bei Atari vier Tongeneratoren zur Verfügung (andere Computer besitzen höchstens drei). Außerdem lassen sich die Sounds unkompliziert programmieren.

```
REM PUNKTVERLUST/FEHLSCHLAG
10 FOR X=0 TO 3
20 READ A,B,C
30 SOUND 0,A,14,B
40 FOR Z=0 TO C:NEXT Z
50 NEXT X
60 END
70 DATA 30,15,200
80 DATA 40,8,300
90 DATA 50,10,200
100 DATA 60,6,300
```

Mit READ A,B,C werden Ton, Lautstärke und Abspielzeit eingelesen. Die Werte für die Melodie stehen in den DATA-Zeilen. Mit Zeile 30 wird die Melodie abgespielt. 0 ist der erste Tongenerator, A der aktuelle Ton, 14 der Verzerrungsgrad (14 ist „reiner Ton"), B ist die aktuelle Lautstärke. Zeile 40 schafft eine künstliche Pause, deren Dauer durch den Wert C angegeben wird (200 oder 300). Während dieser Pause wird der Ton gespielt.

```
REM GEFAHR/ALARM
10 FOR Y=1 TO 3
20 FOR X=200 TO 50 STEP -2
30 SOUND 0,X,14,15
40 NEXT X
50 NEXT Y
```

Die Töne 200 bis 50 werden in schneller Abfolge aufgerufen, wobei jeder zweite Ton übersprungen wird (STEP −2). Durch die FOR ... NEXT-Schleife in Zeile 10 und 50 wird der Abspielvorgang dreimal wiederholt.

```
REM FALLENDES OBJEKT
10 FOR X=0 TO 255
20 SOUND 0,X,10,15
30 NEXT X
```

Die Struktur dieses Programms wurde am Anfang bei den Prinzipien erklärt.

```
REM ZUSAMMENPRALL
10 SOUND 0,25,2,15
20 FOR Z=0 TO 300:NEXT Z
```

```
REM FAHRZEUG
10 SOUND 0,170,12,15
20 FOR Z=0 TO 500:NEXT Z
```

```
REM EXPLOSION
10 SOUND 0,25,0,15
20 FOR Z=0 TO 500:NEXT Z
```

Bei Atari wirken die Werte von Ton und Verzerrungsgrad in „geheimnisvoller" Weise zusammen. So kann es vorkommen, daß bei Ton 0 und Verzerrungsgrad 10 gar nichts zu hören ist. Das gleiche gilt für Ton 185 bei Verzerrungsgrad 12. Andererseits können allein durch den richtigen Ton beim richtigen Verzerrungsgrad Geräusche erzeugt werden, die sonst nur durch umständliches Aufrufen mehrerer Tongeneratoren möglich werden. Die drei kurzen Listings oben sind Beispiele dafür. Weitere interessante Effekte erreichen Sie mit folgenden Werten (Angabe jeweils Ton, Verzerrungsgrad): 45,2 – 40,12 – 15,4 – 10,4 – 170,12 – 140,12 – 125,12 – 20,0 – 15,0.

Dragon 32

Zum Dragon gibt es eine schlechte und eine gute Nachricht. Die schlechte ist: Er besitzt nur einen Tongenerator. Die gute Nachricht: Man kann Musik bequemer programmieren als bei anderen Computern. Bei Verwendung des PLAY-Befehls kann man anstelle von Zahlenwerten die gewünschte Note direkt eingeben; zum Beispiel „E" oder „G" und so weiter. Zusätzlich hat der Dragon noch das Sound-Statement, mit dem verschiedene Effekte erzielt werden können.

```
REM PUNKTVERLUST/FEHLSCHLAG
10 FOR X=160 TO 130 STEP -4
20 SOUND X,1
30 NEXT X
```

Mit einer FOR ... NEXT-Schleife werden die Frequenzen von 160 bis 130 aufgerufen, wobei jeweils vier übersprungen werden (STEP −4). Sie können hier auch einen anderen Wert eingeben. Je mehr Töne übersprungen werden, um so schneller und abgehackter wirkt der Sound-Effekt. Mit SOUND X,1 wird der aktuelle Ton in der Zeitspanne 1 gespielt (1/16 Sek).

```
REM PUNKTGEWINN/SIEGESMELODIE
10 PLAY "L16GEEL8AG"
```

L 16 bedeutet 1/16-Note (Note mit zwei „Fähnchen"). G E E sind die Töne, die als 1/16-Note gespielt werden sollen. L 8 bedeutet 1/8-Note. A G sollen als 1/8-Noten gespielt werden.

```
REM NIEDERLAGE
10 PLAY "L4DDL8DL4DFL8EL4EDL8DL4D"
```

Diese Programmzeile ist nach dem gleichen Prinzip aufgebaut wie im vorhergehenden Beispiel. Anstelle einer Siegesfanfare wird hier ein Trauermarsch gespielt.

TI-99/4A

```
REM FEHLSCHLAG/PUNKTVERLUST
10 FOR X=550 TO 250 STEP -50
20 CALL SOUND (1,X,0)
30 NEXT X
```

Beim TI werden die Töne in Form ihrer Frequenzen aufgerufen. In Zeile 10 werden die Frequenzen 550 bis 250 angesprochen, wobei immer 50 übersprungen werden (STEP −50). Mit CALL SOUND in Zeile 20 wird bei der Tondauer 1 die aktuelle Note X in der Lautstärke 0 (= größte Lautstärke) ausgegeben.

```
REM FEHLSCHLAG/PUNKTVERLUST
10 CALL SOUND (100,220,0)
20 CALL SOUND (100.175,0)
30 CALL SOUND (200,-3,0)
```

Drei Töne werden aufgerufen, wobei der letzte ein periodisches Rauschen ist.

```
REM GEFAHR/ALARM
10 FOR Y=1 TO 3
20 FOR X=1500 TO 2500 STEP 250
30 CALL SOUND (-1000,X,0,-1,0)
40 NEXT X
50 NEXT Y
```

Mit X=1500 TO 2500 werden relativ hohe Frequenzen aufgerufen. Mit −1000 in Zeile 30 wird erreicht, daß die Abfolge der Töne sehr schnell ist. Zusätzlich zum aktuellen Ton X und Lautstärke 0 wird mit −1 ein periodisches Rauschen aufgerufen, damit das Ganze auch hübsch „gefährlich" klingt.

Scan: S. Höltgen


```
REM PUNKTGEWINN/SIEGESMELODIE

10 CALL SOUND (100,392,0)
20 CALL SOUND (100.392,0,330,0)
30 CALL SOUND (100,392,0,330,0)
40 CALL SOUND (200,440,0,392,0)
50 CALL SOUND (200,392,0,330,0)
```

Hier wird die gleiche Siegesmelodie gespielt wie im Listing beim ZX Spectrum. Allerdings werden hier beim TI zwei Tongeneratoren angesprochen, wodurch die Melodie voller klingt.

Colour-Genie EG 2000

```
REM FEHLSCHLAG/PUNKTVERLUST

10 FOR Y=4 TO 3 STEP -1
20 FOR X=7 TO 1 STEP -.3
30 PLAY (1,Y,X,15)
40 NEXT X
50 NEXT Y
60 PLAY (1,Y,X,0)
```

Y ist die Oktave, X der Ton, der innerhalb der Oktave gespielt werden soll. Dummerweise haben die Konstrukteure die Werte für die Halbnoten (Cis, Fis und so weiter) hinter die Ganznoten gelegt. Deshalb ist C = 1, Cis = 8. Sinnvoller wäre es gewesen, C = 1, Cis = 2 vorzusehen. Die Zeile 10 könnte dann lauten FOR Y=12 TO 1. Durch den erzwungenen Verzicht auf die Halbnoten klingt der „Fehlschlag"-Sound nicht so „traurig" wie in den anderen Programmen.

```
REM GEFAHR/ALARM

10 FOR Z=0 TO 2
20 FOR Y=2 TO 4
30 FOR X=1 TO 7
40 PLAY (1,Y,X,15)
50 NEXT X
60 NEXT Y
70 NEXT Z
80 PLAY (1,Y,X,0)
```

Dieses Programm ist genauso strukturiert wie das ZX-Spectrum-Programm. Der Unterschied besteht darin, daß neben den Tönen und der dreimaligen Wiederholung noch die Oktaven aufgerufen werden.

```
REM FALLENDES OBJEKT

10 FOR X=0 TO 255
20 SOUND 0,X
30 SOUND 6,0
40 SOUND 7,8
50 SOUND 8,15
60 NEXT X
70 SOUND 8,0
```

Bei diesem Programm wird nicht mit dem PLAY-Befehl, sondern mit SOUND gearbeitet, womit man die Tongeneratoren fast ebenso differenziert steuern kann wie beim Commodore. In Zeile 10 werden alle verfügbaren Töne aufgerufen. Zeile 20 läßt den aktuellen Ton X auf dem ersten Tonkanal 0 erklingen. Zeile 30 sorgt für den „reinen" Klang (alles, was über 6,0 hinausgeht, verursacht Rauschen; gröbstes Rauschen ist 6,31). Zeile 40 aktiviert den Befehl in Zeile 30. Mit 8,15 in Zeile 50 wird die größte Lautstärke erzeugt. SOUND 8,0 in Zeile 70 ist notwendig, um den Tonkanal wieder abzuschalten.

```
REM ZUSAMMENPRALL/EXPLOSION

10 FOR R=20 TO 31 STEP .2
20 SOUND 0,10
30 SOUND 6,R
40 SOUND 8,15
50 NEXT R
60 FOR L=15 TO 2 STEP -.4
70 SOUND 8,L
80 FOR Z=0 TO 20:NEXT Z
90 NEXT L
100 SOUND 8,0
```

Zeile 10: R steht als Abkürzung für Rauschen. Die Verzerrungsgrade von 20 bis 31 werden aufgerufen; STEP .2 sorgt für eine notwendige Verzögerung. Die Frequenz 10 wird in Zeile 20 als Grundfrequenz für das Rauschen genommen. 6,R in Zeile 30 gibt den aktuellen Verzerrungsgrad aus. Ab Zeile 60 wird die Explosion durch Verminderung der Lautstärke zum Abklingen gebracht. Die Werte können beliebig verändert werden.

Commodore 64

Die Basis-Adresse des Sound-Chip ist 54272. Da der Computer bei Variablen nur die ersten beiden Buchstaben erkennt, wurde in den Listings die Variable BASIS mit BA abgekürzt. HI ist die Frequenz Hi-Byte. Auf LO-Byte-Wert kann bei der Programmierung von Sound-Effekten zum Teil verzichtet werden.

```
REM FALLENDES OBJEKT

10 BASIS=54272
20 POKE BA+24,15
30 POKE BA+6,15*16
40 FOR HI=80 TO 10 STEP -1
50 POKE BA+1,HI
60 POKE BA+4,17
70 FOR X=0 TO 10:NEXT X
80 NEXT HI
90 POKE BA+24,0
```

In Zeile 20 wird die größte Lautstärke erzeugt. 15∗16 in Zeile 30 sorgt für einen Dauerton ohne Anschlag und Abschwellen. In Zeile 40 werden die Frequenzen festgelegt, die durchlaufen werden sollen. Mit 17 in Zeile 60 wird eine Dreieckswelle erzeugt. Der Wert kann durch 33 ersetzt werden. Die FOR... NEXT-Schleife in Zeile 70 sorgt für eine notwendige Verzögerung, da der Ablauf des Programms sonst so schnell vonstatten ginge, daß die Töne kaum hörbar wären. Zeile 90 schaltet den Tongenerator aus.

```
REM FEHLSCHLAG/PUNKTVERLUST

10 BASIS=54272:LAUT=15
20 FOR HI=60 TO 48 STEP -2
30 POKE BA+24,LAUT
40 POKE BA+6,15*16
50 POKE BA+1,HI
60 POKE BA+4,17
70 FOR X=0 TO 100:NEXT X
80 LAUT=LAUT -1
90 NEXT HI
100 BA+24,0
```

Dieses Programm ist im Prinzip genauso aufgebaut wie im vorhergehenden Beispiel. Allerdings wird hier zusätzlich zum Aufruf der Frequenzen in jedem Durchlauf die Lautstärke etwas reduziert.

```
REM GEFAHR/ALARM

10 BASIS=54272
20 FOR Y=1 TO 3
30 LAUT=10
40 FOR HI=70 TO 100 STEP 2
50 POKE BA+24,LAUT
60 POKE BA+6,15*16
70 POKE BA+1,HI
80 POKE BA+4,33
90 LAUT=LAUT +1
100 IF LAUT >14 THEN LAUT=15
110 NEXT HI
120 NEXT Y
130 POKE BA+24,0
```

Der Programmaufbau entspricht dem im Listing „Fallendes Objekt". Der Unterschied besteht darin, daß die Anfangslautstärke 10 beträgt und im Verlauf des Ertönens auf 15 anschwellt. Mit Y=1 TO 3 wird erreicht, daß das Warnsignal dreimal hintereinander erklingt. Zeile 100 ist notwendig, da jeder Lautstärkewert über 15 eine Fehlermeldung verursachen würde.

Tips zum Einbau der Sound-Effekte

Sie können sämtliche Listings in Form von Unterprogrammen in Ihre bestehenden Spiele einbringen und mit GOSUB, wo immer sie wollen, aufrufen. Wichtig ist, daß Sie keine Variablen-Bezeichnungen verwechseln, die sich vielleicht in Ihrem Programm befinden. Außerdem müssen Sie am Ende der Sound-Listings den Rechner mit RETURN ins Programm zurückführen.

Alfred Görgens

Scan: S. Höltgen

TI 99/4A – Mit Volldampf voraus

Die Beerdigung fand nicht statt. Auch nach dem endgültigen Produktions-Stop für den TI 99/4A ist noch eine Menge nützlichen Zubehörs auf dem Markt. Der Totgesagte bleibt recht lebendig

Scan: S. Höltgen

1 V.24-Anschluß, zwei Meter lang von Atronic, Hamburg, Preis circa 60 Mark.
2 Problemlose Verbindung zu handelsüblichen Kassettenrecordern, von Dynamics, Hamburg, Preis etwa 40 Mark.
3 Anschluß für zwei Joysticks von Fremdherstellern, von Axel Rincke, München, Preis rund 30 Mark.
4 Kompakter Joystick-Adapter von Dynamics, Hamburg, für etwa 20 Mark.
5 Noch ein Anschluß für zwei Joysticks, von Axel Rincke, München, rund 25 Mark.
6 Verlängerungskabel für alle Joystick- oder Rekorderkabel, 2,5 Meter lang, von Axel Rincke, München, Preis circa 10 Mark.
7 Vielseitiger Printer/Plotter mit Centronics-Schnittstelle, vier Farben und 114,5 Millimeter Papierbreite. Arbeitet nach dem Ball-Point-Prinzip und kostet etwa 450 Mark. Von Mechatronic, Sindelfingen.
8 Peripherie-Box mit Disketten-Controller und einem Disketten-Laufwerk (insgesamt acht Steckplätze für Karten) für knapp 1200 Mark. Vertrieb: Computer-Hüsli, München.
9 Dynamische Speichererweiterung 32 KByte RAM, mit Centronics-Schnittstelle, Speichererweiterung 64 KByte RAM mit Centronics-Schnittstelle und 32 K-CMOS-Speichererweiterung. Keine verbindliche Preisangabe, von Atronic, Hamburg.
10 Diskettenstation mit BASF-Laufwerk, doppelte Dichte (500 KByte unformatiert), Diskette beidseitig zu beschreiben. Von Computer-Hüsli, München, zum Preis von etwa 1100 Mark.
11 Diskettenstation mit Disk-Manager und -Controller für 2350 Mark, 32 KByte Speicher für 545 Mark, Schnittstellenbox mit zwei RS 232 und einer Parallel-Schnittstelle 210 Mark, von Schuh & Co, Germering.
12 Matrix-Drucker Epson RX 80, angeboten von Computer-Hüsli, München, für circa 1000 Mark.
13 Akustik-Koppler Epson CX 21 von Computer-Hüsli, München, für etwa 700 Mark, die Treiber-Software dazu – nötig für den Anschluß des TI 99/4A kostet knapp 150 Mark.
14 Sprach-Synthesizer Original TI, vertrieben von Computer-Hüsli, München, Preis circa 170 Mark.
15 Anschlußkabel für Rekorder von Atronic. Preis etwa 50 Mark.
16 Anschluß für zwei Kassettenrekorder gleichzeitig (auf dem zweiten kann nur abgespeichert werden), von Axel Rincke, München. Preis etwa 45 Mark.
17 Verbindungskabel für Stereorecorder mit eigenem Verstärker und regelbarem Kopfhörerausgang von Rincke. Preis etwa 35 Mark.
18 Anschlußkabel für die Dateneingabe per Walkman von Rincke. Es kostet etwa 7 Mark.

Scan: S. Höltgen

1 Software von Apesoft: Apesoft Expanded Grafic BASIC – superschnelle Grafikerzeugung mit 16 Vorder- und Hintergrundfarben, 40 Befehle, Einzelpunkt-Adressierung. Zu beziehen über Computer-Hüsli, München, für rund 200 Mark (Kassette) oder 220 Mark auf Diskette. Cerberus – Videogame auf Diskette. Eine Jagd durch die Hölle für 120 Mark. Außerdem TI-99-Specials – alle Programme der Bücher auf Diskette, von etwa 60 bis 40 Mark; Hilfsprogramme für Druckerunterstützung, Sortieren und mathematische Funktionen für jeweils ca. 100 (Kassette) oder 120 Mark (Diskette).

2 Superprogramme von Dynamics, Hamburg: Ordnungsliebenden empfehlen sich die diversen Archive. Für 200 Adressen (bei RAM-Speichererweiterung) oder Video-Bänder. Sie kosten jeweils etwa 70 Mark. Für Bücher oder Schallplatten knapp 50 Mark. Professionellen Ansprüchen dürften die Programme „Lagerkartei" und „Data-Bank" (je circa 100 Mark) entsprechen. Folgende Programme verlangen nach Extended-BASIC: Der Vokabel-Trainer, der für knapp 50 Mark 200 Vokabeln speichert, das „City-Quiz" – es kostet rund 40 Mark und fragt nach 28 europäischen Hauptstädten – sowie eine Reihe von Spielen: So etwa "Darts" „Zentoria", „Digger" und „Monoco". Jedes Spiel kostet etwa 40 Mark, alle Dynamics-Programme sind auf Kassette gespeichert.

3 Literatur zum Thema TI-99 gibt es reichlich – hier eine aktuelle Auswahl: Das deutsche Handbuch Editor/Assembler von Wickanderware, Berlin, ist für etwa 100 Mark erhältlich, „Spielen, Lernen, Arbeiten mit dem TI-99/4A" ist im Sybex-Verlag (Düsseldorf) erschienen und kostet etwa 30 Mark. „Tips & Tricks" vermittelt ein Buch von Data Becker (Düsseldorf) für rund 50 Mark. Ebensoviel kostet das Buch „Programme für den TI-99/A" aus dem Hofacker-Verlag, Holzkirchen. Texas Instruments selbst gibt eine Reihe von umfangreichen Broschüren heraus, unter anderem „TI-BASIC/Extended-BASIC für Anfänger und Fortgeschrittene" (knapp 50 Mark) und die Programmierhandbücher für fortgeschrittene Benutzer „Special 1" und „Special 2". Ihr Preis beträgt etwa 50 beziehungsweise 55 Mark. *– hs*

Scan: S. Höltgen


```
9060 DATA "  ","  }","  |","  {
","  }{","  |{","  {{","}{{","|{{"
9070 DIM B(6): RESTORE 9090: FOR
 N=1 TO 6
9080 READ B(N): NEXT N: RETURN
9090 DATA .5,.5,.4,.28,.15,.10
9999 SAVE "WALL" LINE 0: POKE 23
736,181: SAVE "WALL"CODE 31744,1
024
```

LISTING 2

```
 5 REM Programm zum Verlegen
       des Zeichensatzes
10 CLEAR 31743
20 FOR i=0 TO 775
30 POKE 31744+i,PEEK (15616+i)
40 NEXT i
50 POKE 23607,123
```

LISTING 3

```
 5 REM Programm zum Eingeben
       des Zeichensatzes
 6 REM SAVEN mit SAVE " "
       CODE 31744,1024
10 INPUT "Adresse=";a
20 INPUT (a);": "; LINE a$
30 IF a$="e" THEN GO TO 90
40 IF a$="a" THEN GO TO 10
50 POKE a,VAL a$
60 PRINT a;": ";a$
70 LET a=a+1
80 GO TO 20
90 POKE 23607,123
```

LISTING 4

```
32216 :  247  227  201  28
32220 :  56   114  230  54
32224 :  247  247  195  24
32228 :  218  218  218  24
32232 :  247  135  39   112
32236 :  56   155  206  4
32240 :  62   127  73   73
32244 :  62   34   54   28

32264 :  56   116  120  56
32268 :  16   56   56   56
32272 :  56   56   16   16
32276 :  16   16   16   24
32280 :  28   46   30   28
32284 :  8    28   28   28
32288 :  28   28   8    8
32292 :  8    8    8    24
32296 :  56   116  120  56
32300 :  16   120  251  190
32304 :  184  184  56   24
32308 :  20   34   68   102
32312 :  28   46   30   28
32316 :  8    30   223  125
32320 :  29   29   28   24
32324 :  40   68   34   102
32328 :  56   116  120  56
32332 :  16   120  251  190
32336 :  56   56   24   20
32340 :  24   112  80   56
32344 :  28   46   30   28
32348 :  8    30   223  125
32352 :  28   28   24   40
32356 :  24   14   10   28
32360 :  0    0    56   116
32364 :  120  56   16   60
32368 :  58   57   56   56
32372 :  16   8    18   24
32376 :  0    0    28   46
32380 :  30   28   8    60
32384 :  92   155  28   28
32388 :  8    16   8    24
32392 :  195  66   126  153
32396 :  255  195  126  102
32400 :  66   126  219  153
32404 :  255  66   60   195
32408 :  36   102  255  153
32412 :  189  231  60   195
32416 :  247  247  247  0
32420 :  254  254  254  0
32424 :  247  247  247  0
32428 :  255  255  255  0

32448 :  56   84   124  56
32452 :  16   56   124  186
32456 :  186  186  40   40
32460 :  40   40   40   108
32464 :  186  254  124  40
32468 :  40   40   40   108
32472 :  62   62   0    62
32476 :  62   0    62   62
32480 :  0    0    0    62
32484 :  62   0    62   62
32488 :  0    0    0    0
32492 :  0    0    62   62
```

Miner

In einer Mine wird nach Gold gegraben. Sandstein, Granit, Wassereinbrüche und Treibsand behindern die Arbeiten auf dem TI-99/4A mit Extended-BASIC

Der erste Weg der weißen Spielfigur führt zum Aufzug, der automatisch in die Mine einfährt. Bei der Höhe, an der gegraben werden soll, muß „X" gedrückt werden. Jetzt kann zu einer vermuteten Goldstelle hingefahren werden.

Nicht auf Granit beißen!

Es kann sich jedoch herausstellen, daß man sich geirrt hat und dort Sandstein, Granit, einen Wassereinbruch oder Treibsand vorfindet.

Bei Treibsand wird der Schacht zugeschüttet. Ein Granit-Block kann mit „B" weggesprengt oder mit „N" umgangen werden. Bei Wassereinbruch kann mit „P" das Wasser abgepumpt oder mit „N" umgangen werden. Bei allen Aktivitäten wird Geld und Energie verloren.

In der oberen Bildschirmzeile sind alle für den Spielablauf wichtigen Informationen angezeigt: wieviel Geld zur Verfügung steht, die Anzahl der Unzen Gold, wieviel Energie noch vorhanden ist.

Scan: S. Höltgen

Neue Kraftreserven holen!

Nähert sich die Energie dem Nullpunkt, so ist es ratsam, in den Aufzug zu steigen, mit „E“ nach oben zu fahren und mit „S“ in ein Gebäude zu gehen. Dort kann die Energie wieder auf den Wert 250 heraufgesetzt werden, und man hat erneut Kraft zum Arbeiten. Ein Spiel endet mit der Erschöpfung der Energie oder mit einem Sturz in den Schacht.

Alexander Kraus

```
100 REM ****     MINER    ****          ****
MADE IN 1984****           **** 8061 BYTES
****
110 ON ERROR 2210
120 PUNKT=0 :: MONEY=1000 :: CALL CLEAR
:: ENERGIE=1000
130 CALL SCREEN(2):: FOR N=1 TO 9 :: CAL
L COLOR(N,12,2):: NEXT N :: GOSUB 1960
140 PUNKT=0 :: MONEY=1000 :: CALL CLEAR
:: ENERGIE=250
150 CALL CHAR(33,"38B992FC3838286C")
160 MONEY=1000
170 CALL CHAR(96,"3C7E7E7E3C18183C")
180 CALL CHAR(104,"0F1F3F7F66667E7EF0F8F
CFE66667E7E")!HAUS
190 CALL CHAR(112,"20408080FF80402004020
101FF0102043C4A8908080808081010101010915
23C")
200 CALL CHAR(120,"FF818181FF81FFFF819
99BBDFF99FF10101010101010")
210 CALL CHAR(128,"0C12619E21C01EE1")
220 CALL CHAR(136,"08181C3E3F7FFFFF")
230 CALL CHAR(137,RPT$("00",8),138,"FF00
FF00")
240 CALL CHARPAT(ASC("("),A$,ASC(")"),B$
):: CALL CHAR(91,A$,93,B$)
250 CALL CLEAR
260 CALL COLOR(1,15,1,2,2,11,9,3,1,10,9,
1,11,16,2,12,15,1,13,16,5,14,9,11)
270 CALL SCREEN(2)
280 CALL HCHAR(3,1,138,32)
290 CALL HCHAR(4,1,137,20*32)
300 CALL HCHAR(2,1,96,15)
310 CALL VCHAR(1,30,122,23):: CALL HCHAR
(23,1,138,32)
320 CALL HCHAR(1,30,122):: CALL HCHAR(2,
30,120)
330 CALL HCHAR(2,20,104):: CALL HCHAR(2,
21,105)
340 CALL VCHAR(5,29,32,3):: CALL VCHAR(1
0,29,32,3):: CALL VCHAR(15,29,32,3):: CA
LL VCHAR(20,29,32,3)
350 CALL HCHAR(9,3,138,29):: CALL HCHAR(
14,3,138,29):: CALL HCHAR(18,3,138,29)
360 FOR N=1 TO 50 :: CALL SOUND(-100,N*2
0+100,20)
370 RANDOMIZE
380 B=INT(RND*27)+2 :: A=INT(RND*19)+4
390 CALL GCHAR(A,B,C):: IF C<>137 THEN 3
70
400 CALL HCHAR(A,B,46):: PUNKT=PUNKT+1 :
: NEXT N
410 A=2 :: B=25
420 CALL HCHAR(A,B,33)
430 CALL KEY(3,K,S):: IF S=0 THEN 430
440 A1=A :: B1=B
450 IF MONEY>30000 THEN 2130
460 IF K=68 THEN B=B+1
470 IF K=83 THEN B=B-1
480 IF K=69 THEN A=A-1
490 IF K=88 THEN A=A+1
500 CALL GCHAR(A,B,C):: IF C=137 THEN 85
0
510 IF C=120 THEN 690
520 IF C=128 THEN 1750
530 IF C=122 THEN 1100
540 IF C=136 THEN 1300
550 IF C=138 THEN A=A1 :: B=B1 :: GOTO 4
20 ELSE 560
560 IF C=46 THEN 1210
570 IF C=105 THEN 1820
580 CALL HCHAR(A1,B1,32)
590 GOTO 420
600 A=A+1
610 FOR N=A-1 TO 2 STEP -1
620 CALL HCHAR(N,30,121):: CALL HCHAR(N+
1,30,122)
630 A=A-1 :: CALL SOUND(-100,N*50+100,5)
640 CALL KEY(3,K,S)
650 IF K=83 THEN GOTO 810 ELSE IF K=88 T
HEN 700
660 NEXT N
670 CALL HCHAR(N+1,30,120):: CALL HCHAR(
A,B,33)
680 GOTO 420
690 B=B1
700 CALL HCHAR(A1,B1,32):: CALL HCHAR(A,
B+1,121):: MONEY=MONEY-10
710 IF A=22 THEN 600
720 FOR N=A+1 TO 22
730 A=A+1
740 CALL HCHAR(N,30,121):: CALL HCHAR(N-
1,30,122)
750 CALL SOUND(-100,N*50+100,5)
760 CALL KEY(3,K,S)
770 IF K=83 THEN 830 ELSE IF K=69 THEN 6
00
780 NEXT N
790 CALL HCHAR(N-1,30,120):: CALL HCHAR(
A,B,33)
800 GOTO 420
810 CALL GCHAR(A,B,C):: IF C<>32 THEN 66
0
820 GOTO 840
830 CALL GCHAR(A,B,C):: IF C<>32 THEN 78
0
840 CALL HCHAR(N,30,120):: CALL HCHAR(A,
B,33):: GOTO 420
850 REM PICKEL
860 ENERGIE=ENERGIE-5
870 GOSUB 2360
880 IF K=69 THEN 930
890 IF K=68 THEN 980
900 IF K=88 THEN 1020
910 IF K=83 THEN 1060
920 GOTO 850
930 FOR N=1 TO 10
940 CALL HCHAR(A,B,114):: CALL SOUND(-1,
110,5):: CALL HCHAR(A,B,113)
950 NEXT N
960 MONEY=MONEY-10
970 CALL HCHAR(A1,B1,32):: GOTO 420
980 FOR N=1 TO 10
990 CALL HCHAR(A,B,113):: CALL SOUND(-1,
110,5):: CALL HCHAR(A,B,114)
1000 NEXT N
1010 GOTO 960
1020 FOR N=1 TO 10
```

Scan: S. Höltgen


```
1030 CALL HCHAR(A,B,115):: CALL SOUND(-1
,110,5):: CALL HCHAR(A,B,113)
1040 NEXT N
1050 GOTO 960
1060 FOR N=1 TO 10
1070 CALL HCHAR(A,B,112):: CALL SOUND(-1
,110,5):: CALL HCHAR(A,B,114)
1080 NEXT N
1090 GOTO 960
1100 CALL HCHAR(A1,B1,32)
1110 CALL HCHAR(24,30,64)
1120 FOR N=1 TO 20 :: CALL SOUND(-100,11
0*(21-N),10):: NEXT N
1130 CALL CLEAR
1140 CALL VCHAR(5,6,137,10):: CALL HCHAR
(8,3,137,7)
1150 DISPLAY AT(8,20)SIZE(9):USING "$ ##
#####":MONEY
1160 DISPLAY AT(16,1):"AMEN"
1170 DISPLAY AT(18,1):"SIE SIND IN DEN A
UFZUG-      SCHACHT GEFALLEN UND SIND   T
 O T":"------"
1180 DISPLAY AT(22,1):"NOCH EIN SPIEL? J
/N"
1190 CALL KEY(3,K,S):: IF S=0 THEN 1190
1200 IF K=74 THEN RUN ELSE IF K=78 THEN
END ELSE GOTO 1190
1210 RANDOMIZE
1220 PUNKT=PUNKT-1
1230 Z=INT(RND*5)+1
1240 ON Z GOTO 1250,1300,1420,1510,1590
1250 DISPLAY AT(24,1):"SANDSTEIN"
1260 MONEY=MONEY-5 :: ENERGIE=ENERGIE-6
:: GOSUB 2360
1270 CALL HCHAR(A1,B1,32):: CALL HCHAR(A
,B,33)
1280 CALL SOUND(-1,500,5)
1290 FOR N=1 TO 350 :: NEXT N :: DISPLAY
 AT(24,1):"" :: GOTO 420
1300 DISPLAY AT(24,1):"GRANIT"
1310 CALL HCHAR(A,B,136)
1320 CALL KEY(3,K,S):: IF S=0 THEN 1320
1330 IF K=66 THEN 1340 ELSE IF K=78 THEN
 A=A1 :: B=B1 :: DISPLAY AT(24,1):"" ::
GOTO 420 ELSE 1320
1340 CALL SOUND(750,-5,20)
1350 REM
1360 FOR N=1 TO 250 :: NEXT N :: CALL HC
HAR(A,B,32)
1370 MONEY=MONEY-30 :: ENERGIE=ENERGIE-8
 :: GOSUB 2360
1380 FOR N=1 TO 30 STEP 2
1390 CALL SOUND(100,-6,N)
1400 NEXT N :: DISPLAY AT(24,1):""
1410 CALL HCHAR(A,B,32):: CALL HCHAR(A1,
B1,32):: GOTO 420
1420 REM GOLD
1430 FOR N=1 TO 10
1440 CALL SOUND(-100,-2,5):: CALL SOUND(
-100,-1,5):: CALL SOUND(-100,-3,5):: NEX
T N
1450 CALL HCHAR(A,B,71)
1460 Z=INT(RND*4)+1 :: DISPLAY AT(24,1):
Z;"UNZE[N] GOLD"
1470 GOLD=GOLD+Z :: ENERGIE=ENERGIE-5
1480 GOSUB 2360
1490 FOR N=1 TO 350 :: NEXT N :: DISPLAY
 AT(24,1):""
1500 CALL HCHAR(A1,B1,32):: GOTO 420
1510 IF RND>.8 THEN 1520 ELSE A=A1 :: B=
B1 :: PUNKT=PUNKT+1 :: GOTO 420
1520 FOR N=1 TO 10 :: CALL SOUND(-10,500
,5):: CALL SOUND(-10,600,5):: NEXT N
1530 PUNKT=PUNKT+1 :: ENERGIE=ENERGIE-8
:: GOSUB 2360
1540 DISPLAY AT(24,1):"WASSEREINBRUCH"
1550 FOR N=A-2 TO A+2
1560 FOR X=1 TO 29
1570 CALL GCHAR(N,X,C):: IF C=32 THEN CA
LL HCHAR(N,X,128)ELSE 1580
1580 NEXT X :: NEXT N :: A=A1 :: B=B1 ::
 DISPLAY AT(24,1):"" :: GOTO 420
1590 CALL SOUND(4000,-5,25)
1600 DISPLAY AT(24,1):"TREIBSAND"
1610 A2=A-2 :: IF A2<3 THEN A2=A
1620 A3=A+2 :: IF A3>22 THEN A3=A
1630 FOR N=A2 TO A3
1640 FOR X=1 TO 29
1650 CALL GCHAR(N,X,C):: IF C=32 THEN CA
LL HCHAR(N,X,137)
1660 NEXT X :: NEXT N
1670 DISPLAY AT(24,1):""
1680 RANDOMIZE :: IF RND>.94 THEN 1700
1690 A=A1 :: B=B1 :: PUNKT=PUNKT+1 :: GO
TO 420
1700 DISPLAY AT(24,1):"ALLES GOLD WEG"
1710 GOLD=0 :: GOSUB 2360
1720 FOR N=1 TO 250
1730 NEXT N :: DISPLAY AT(24,1):""
1740 GOTO 1690
1750 DISPLAY AT(24,1):"PUMPEN ? JA=P"
1760 CALL KEY(3,K,S):: IF S=0 THEN 1760
1770 IF K=80 THEN 1780 ELSE IF K=78 THEN
 A=A1 :: B=B1 :: DISPLAY AT(24,1):"" ::
GOTO 420 ELSE 1760
1780 FOR N=1 TO 10
1790 CALL HCHAR(A,B,79):: CALL SOUND(1,4
00,5):: CALL HCHAR(A,B,88):: NEXT N
1800 MONEY=MONEY-50 :: ENERGIE=ENERGIE-1
0 :: GOSUB 2360
1810 CALL HCHAR(A1,B1,32):: DISPLAY AT(2
4,1):"" :: GOTO 420
1820 FOR N=1 TO GOLD
1830 CALL SOUND(-10,100+110*N,5)
1840 NEXT N
1850 MONEY=MONEY+911.72*GOLD :: GOLD=0
1860 ENERGIE=250 :: GOSUB 2360
1870 IF PUNKT=0 THEN 1890
1880 A=A1 :: B=B1 :: GOTO 420
1890 CALL CLEAR
1900 FOR N=1 TO 10 :: CALL SOUND(-100,N*
110,N*2):: NEXT N :: FOR N=10 TO 1 STEP
-1 :: CALL SOUND(-100,N*110,N*2):: NEXT
N
1910 DISPLAY AT(5,1):USING "$ ######":MO
NEY
1920 DISPLAY AT(10,1):"GRATULIERE":"SIE
HABEN ALLES GOLD          GEFUNDEN DAS IN
DER MINE WAR"
1930 DISPLAY AT(15,1):"UND NUN ZUR NAECH
STEN"
1940 FOR N=1 TO 800 :: NEXT N
1950 GOTO 270
1960 CALL CLEAR
1970 PRINT TAB(13);"MINER":TAB(13);"----
-"
1980 PRINT : : :"VERSUCHEN SIE, SOVIEL G
OLD  WIE MOEGLICH ZU FINDEN UND  ZWAR HI
NTER DEN PUNKTEN !!"
1990 PRINT :"DABEI LAUERN ABER GEFAHREN
 AUF SIE WIE:"
2000 PRINT "WASSEREINBRUCH":"GRANITFELSE
N":"TREIBSAND"
2010 PRINT : :"BITTE TASTE DRUECKEN!"
2020 CALL KEY(0,K,S):: IF S<=0 THEN 2020
2030 PRINT "TASTENFUNKTIONEN:"
2040 PRINT : :"CURSORTASTEN: BEWEGUNG DE
R              SPIELERFIGUR"
2050 PRINT :"B : GRANITFELSEN WEGSPRENGE
N"
2060 PRINT :"P : WASSER ABPUMPEN"
2070 PRINT :"N : UM VOM GRANIT ODER WAS-
 SER, OHNE ENERGIEVERLUST    WEGZUKOMMEN
"
2080 PRINT :"WENN SIE IM AUFZUG NACH OBE
N WOLLEN   DRUECKEN SIE EINFACH >>E<<,"
2090 PRINT "UM NACH UNTEN ZU KOMMEN
DRUECKEN SIE >>X<<!"
2100 PRINT : : :
2110 PRINT "TASTE DRUECKEN!"
2120 CALL KEY(0,K,S):: IF S<=0 THEN 2120
 ELSE CALL CLEAR :: RETURN
2130 CALL CLEAR
2140 FOR N=1 TO 9 :: CALL COLOR(N,15,2):
: NEXT N
2150 PRINT "SIE HABEN DAS ZIEL DES
SPIELS ERREICHT"
2160 PRINT : :"DAS HEISST, DASS SIE EIN
   LEBEN DES WOHLSTANDS          ERARBEITE
T HABEN"
2170 PRINT : :"SIE BESITZEN";MONEY;"$"
2180 PRINT : : :"NOCH EIN SPIEL? >J< >N<
"
2190 CALL KEY(3,K,S):: IF S<=0 THEN 2190
2200 IF K=ASC("J")THEN 150 ELSE IF K=ASC
("N")THEN 2400 ELSE 2190
```

```
2210 ON ERROR 2210
2220 A=A1 :: B=B1 :: RETURN 500
2230 CALL COLOR(1,8,1)
2240 FOR F=1 TO 250 :: NEXT F
2250 CALL SOUND(-1000,-1,25)
2260 CALL COLOR(1,6,1):: FOR F=1 TO 250
:: NEXT F
2270 CALL SOUND(-1000,-2,25)
2280 CALL COLOR(1,5,1):: FOR F=1 TO 250
:: NEXT F
2290 CALL SOUND(500,110,0,330,0,660,0,-3
,0)
2300 CALL CLEAR :: FOR N=1 TO 10 :: CALL
 COLOR(N,9,2):: NEXT N
2310 PRINT "SIE SIND IM SCHACHT ERSTICKT
"
2320 PRINT : : :"SIE HABEN";MONEY;"$ VER
DIENT": :"WOMIT SIE ALLERDINGS JETZT  ":
 :"NICHTS MEHR ANFANGEN KOENNEN"
2330 PRINT : : : : : : :"NOCH EIN SPIEL?
>J< >N<"
2340 CALL KEY(3,K,S):: IF S<=0 THEN 2340
2350 IF K=ASC("J")THEN RUN 140 ELSE IF K
=ASC("N")THEN 2400 ELSE GOTO 2340
2360 REM
2370 DISPLAY AT(1,1):USING "$ #####  UN:
##  EN: ####":MONEY,GOLD,ENERGIE
2380 IF ENERGIE<=1 THEN 2230
2390 RETURN
2400 DISPLAY AT(12,1)BEEP ERASE ALL:"NA
DANN NICHT!" :: FOR N=1 TO 1000 :: NEXT
N :: END
```

Crazy Jumpman

Ein Profi-Programm von großem, dauerhaftem Unterhaltungswert für den C 64 mit Joystick: Crazy Jumpman erklettert ein Gerüst und versucht zur rettenden Leiter zu kommen

Eine aufwendige Einlese-Routine leistet bei der Eingabe der vielen DATA-Zeilen sehr nützliche Hilfestellung. Sie entdeckt zum Beispiel auch, wenn eine ganze Zeile ausgelassen wurde. Eingabefehler sind mit dieser Routine fast ausgeschlossen.

Geldsäcke auflesen

Nach dem Start meldet sich das Programm mit einem Titelbild. Bei einem Tastendruck fragt es: „Game or Construction Mode?“ Nach Eingabe von „G“ beginnt das Spiel.

Es erscheint ein Gerüst auf dem Bildschirm. Die Aufgabe ist es, dieses Gerüst zu erklimmen, um zur Leiter oben links im Bild zu gelangen. Dabei muß man möglichst viele Geldsäcke (mit $-Zeichen) einsammeln, darf jedoch nicht in Kontakt mit Schlangen oder herunterfallenden Bomben gelangen.

Tut man dies doch, so verliert man eines von vier Männchen. Verloren geht ein Männchen auch bei einem zu weiten Sprung oder zu tiefen Fall.

Man steuert die Spielfigur mit dem Joystick in Port 2 und kann damit nach links und

```
READY.

20 PRINT"
30 PRINT"
40 PRINT"
50 PRINT"
60 PRINT"          CRAZY JUMPMAN
70 PRINT"          _____________
100 PRINT"
110 PRINT"       COPYRIGHT 7.1984 BY
120 PRINT"
130 PRINT"
140 PRINT"       P E T E R   M E N K E
150 PRINT"
160 PRINT"           GRIFTWEG 4
170 PRINT"        2212 BRUNSBUETTEL
190 PRINT"
200 PRINT"
210 PRINT"
220 PRINT"
230 PRINT"           BITTE TASTE DRUECKEN"
240 GET A$:IF A$=""THEN240
250 PRINT""
260 GOSUB1000
270 :
280 :
290 REM **VARIABLE**
300 AN=40960
301 ZI=47615+1
310 NA$="CR.JUMPMAN"
320 :
330 :
340 :
350 REM **** EINLESEROUTINE ****
360 S=AN:Z=1000
370 FORI=1TO16:READ X
```

Scan: S. Höltgen

Frischzellenkur

Zusatz-RAM leistet Erste Hilfe bei Speicherplatzmangel

Wenn ein Computer mit nicht mehr als 16 Kilobyte Arbeitsspeicher ausgestattet ist, stößt man schnell an seine Grenzen. Die Frage nach einer Speichererweiterung wird laut. Wie weit läßt sich ein bestimmter Rechner erweitern, und warum läßt sich nicht jeder auf 64 K aufrüsten?

Hierzu ein paar interessante Details: Prinzipiell bestimmt der im Rechner verwendete Mikroprozessor (CPU) den maximal möglichen Speicherausbau. Die in den Home-Computern verwendeten 8-bit-Mikroprozessoren können in der Regel 65536 (= 2 hoch 16) Speicherzellen oder Adressen ansprechen. Das liegt daran, daß der 40polige Prozessorschaltkreis mit nur 16 Adreßleitungen versehen ist, die sich aus zwei 8-bit-Leitungssträngen zusammensetzen.

Im Raum – bildlich gesprochen – von 64 Kilobyte müssen sowohl der Lese- wie auch der Schreib-Lese-Speicher (ROM und RAM) untergebracht werden. Dies führt grundsätzlich dazu, daß von diesem Adreßraum alle Read-Only-Memory-Speicherteile abgezogen werden. Das sind vor allem das Betriebssystem-ROM (zum Beispiel 8 K), das BASIC-ROM (z. B. 8 K), die ROMs für Zeichengenerator und Systemkontrollprogramme (z. B. 1 + 4 K) und freie ROM-Adressen für Steckmodule (z. B. 8 K).

Map gibt Bescheid

Am Beispiel eines gut dokumentierten Rechners wie des VC 20 läßt sich anhand der sogenannten Memory-Map – der Speicherkarte also – sofort ablesen, welche Adressen für zusätzlichen RAM zur Verfügung stehen. Jeder Home-Computer-Besitzer sollte sich zu seinem Rechner diese Unterlagen besorgen. Sie werden ihm die Frage nach dem maximal möglichen Speicherausbau schnell beantworten. Jedoch ergibt die Summe aller freien Adressen, die für zusätzlichen RAM reserviert sind, noch nicht den für den Anwender frei verfügbaren, maximalen Speicherplatz. Die Sache hat einen Haken: Der Speicherbereich darf nicht unterbrochen sein, für BASIC muß ein zusammenhängender Speicherbereich zur Verfügung stehen. RAM-Teile sind oft im Adreßraum verstreut und können nur über POKE-Befehle genutzt werden. Dies liegt daran, weil nach dem Einschalten des Rechners eine Routine gestartet wird, die einen Speichertest durchführt. Dazu schreibt der Prozessor in alle aufeinanderfolgenden Speicherzellen eine Zahl, liest diese anschließend wieder und findet das Speicherende dort, wo er nicht mehr die bereits bekannte, sondern eine andere Zahl findet. Das ist auch bei jeder kleinen RAM-Unterbrechung der Fall, im Beispiel des VC 20 beim Treffen auf den 0,5-K-Video-RAM im Adreßbereich zwischen 4608 und 4096.

Eine Ausnahme hinsichtlich eines an keiner Stelle unterbrochenen, sauberen RAM-Bereichs bilden die Rechner der neuesten Generation, die allerdings noch nicht auf den Home-Sektor durchgedrungen sind: Rechner mit einem „CLEAN RAM". Bei ihnen stehen dem Anwender volle 64 Kilobyte RAM zur Verfügung.

Die Rechnergeneration unserer Tage hat eben noch nicht alles. Dafür gibt es mittlerweile Speichererweiterungen für fast alle Home-Computer, die etwas schwach auf der Brust sind. Entweder vom Originalhersteller oder von Zubehörfirmen. Die Preise für vergleichbare Kilobytes variieren, weil die verschiedenen Rechner verschiedene Arten von RAMs benötigen. Z80-Rechner geben sich mit den preisgünstigeren dynamischen RAMs zufrieden, 6502 – (auch 6510, 6809 etc.) – Computer brauchen entweder teurere statische RAMs oder zusätzliche Controller-Bausteine. Die teuersten aller RAM-Bausteine sind jene, die sich mit einer Batterie puffern lassen und so ihren Inhalt nicht vergessen: die CMOS-RAMs, vertreten im Sharp PC 1500.

Folgende Aufstellung zeigt, welche Home-Computer mehr Puste nötig haben, und was man dafür ungefähr hinblättern muß:

Atari 600 XL (6502C/16 K):
64 K = 349 Mark

Commodore VC 20 (6502B/3.5 K):
8 K = 135 Mark; 16 K = 150 Mark; 32 K = 180 Mark; 64 K = 290 Mark

Sanyo Laser 110 (Z80/4 K):
Sanyo Laser 210 (Z80/8 K):
für alle Modelle: 16 K = 149 Mark; 64 K = 298 Mark

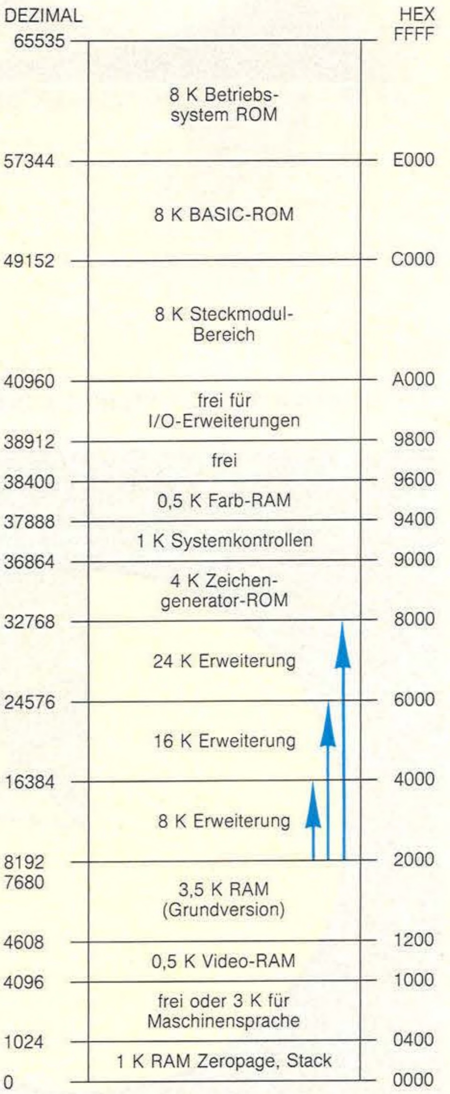
Sharp PC 1500 (LH 5801/2 K):
4 K = 120 Mark; 8 K = 230 Mark; 16 K = 380 Mark; 32 K = 650 Mark

Sinclair ZX 81 (Z80/1 K):
16 K = 100 Mark; 32 K = 150 Mark; 64 K = 200 Mark

Spectrum (Z 80/16 K):
32 K = 120 Mark

Texas Instruments TI 99/4A (TMS 9900/16 K):
(Fa. Atronic) 32 K = 500 Mark

Hans Peter Kroll



Beispiel einer Speicher-Organisation (VC 20-„Hardware-Memory-Map")

Scan: S. Höltgen

dem Bildschirm. Fortgeschrittene gehen zum freien Kritzeln über, Penible bauen ihr Meisterwerk sorgsam und präzise Pixel für Pixel auf.

Dem Anwender stehen 16 Farben zur Verfügung sowie eine Unzahl verschiedener Strichbreiten und -mustern. Zuschaltbarer Lupeneffekt macht die Feinarbeit inmitten der hochauflösenden Grafik zum Vergnügen. Farbenwechsel, Füllen und Radieren in allen nur denkbaren Stadien der Vollendung bereiten ebenfalls keine Probleme, ebensowenig wie die programmunterstützte Zubereitung von einigermaßen perfekten Kreisen, Ellipsen oder Rechtecken. Einigermaßen deshalb, weil sich Linien, die von der Vertikalen oder Horizontalen abweichen, rasterbedingt in dezentem Stufenmuster präsentieren. Punkt- und spiegelsymmetrische Darstellungen liefert die vorbildliche Software gleichfalls ab, sie stellt sogar zwei Zeichenblätter zur Bearbeitung bereit und gestattet das Kopieren von einer Seite auf die andere – der Kreativität sind keine Grenzen gesetzt.

Da außerdem die Möglichkeit besteht, Texte einzublenden, empfiehlt sich das vielseitige Programm auch für Geschäftsleute, die auf die Schnelle Balken- oder Kreisdiagramme erstellen wollen. Konservierungsmöglichkeiten auf Band oder Diskette sind – menügesteuert – ebenfalls gegeben, nicht dagegen die Ansteuerung eines Plotters. Der Hersteller empfiehlt den Griff zum Fotoapparat – ein schwacher Trost. Wer auf Hardcopies verzichten kann, wird zufrieden sein: Die Menüs sind so übersichtlich und zugriffsfreundlich gestaltet, daß auch relativ schwierige und umfangreiche grafische Probleme kinderleicht bewältigt werden können – ohne großes Training, ohne langes Manualwälzen. (Das englische Handbuch ist zudem sehr nutzerfreundlich angelegt). Den Vertrieb des Zeichentabletts hat die Firma Radix-Bürotechnik (Hamburg) übernommen. *hs*

Alles Paletti

Pinseln mit Komfort: Computergrafik ohne Schweißvergießen garantiert ein neues Grafiktablett – ausgestattet mit einer prächtigen Farbpalette und allen denkbaren Raffinessen

Mal mal wieder – dachte sich schon mancher und griff zum Pinsel, um ihn wenig später wieder völlig frustriert wegzuwerfen. Der gekonnte Umgang mit Stiften – ob Farb, ob Filz – setzt ebenfalls beträchtliche Geschicklichkeit voraus: In den wenigsten Fällen entspricht des Endprodukt den Vorstellungen des Künstlers. Der Home-Computer leistet aber nicht nur Denk- und Rechenarbeit, er nimmt auch Hobbykünstlern schier unüberwindliche Schwierigkeiten ab, etwa das Anfertigen einer perfekten Geraden oder eines wirklich runden Kreises.

Falls der Weg zum Kunstwerk allerdings mit Programmierarbeit gepflastert ist, vergeht sogar der Mona Lisa das Lächeln – es dauert eine Ewigkeit. Mit Zeichenhilfen wie Lichtgriffeln oder Grafiktabletts geht es wesentlich flotter voran, auch wenn sich die natürliche Zitteramplitude des Nutzers oft störend auf der Bildschirmdarstellung bemerkbar macht. Programmiermüde Präzisionspinsler greifen dennoch gerne zum Tablett, und schon kommt Kunst auf.

Die neueste elektronische Maltafel nennt sich Super-Sketch, stammt aus Amerika (von Personal Peripherials, Inc.) und kostet knapp 300 Mark mit allem Drum und Dran, also einem Steckmodul und einem Verbindungskabel zu den Rechnern Commodore C 64 oder TI 99/4A. Der Einstieg ins bunte Vergnügen setzt folglich keine Diskettenstation voraus. Das Gerät selbst besticht durch eine verhältnismäßig große nutzbare Arbeitsfläche von 22 mal 19 Zentimetern. (Zum Vergleich: Das Koala-Pad stellt gerade elf Zentimeter im Quadrat zu Zeichenzwecken zur Verfügung).

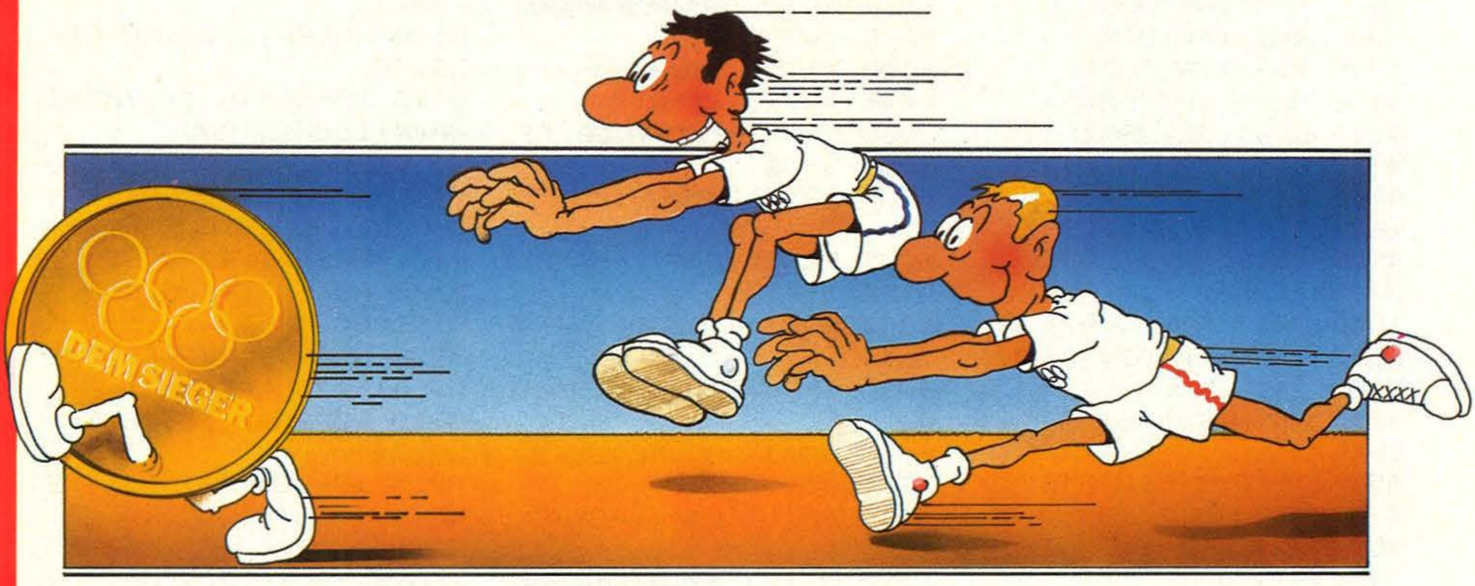
Design aus der Plastik-Ära

Weniger berauschend wirken Design und Material auf den Betrachter: Das Gerät besteht aus weißem Preiswertplastik und vermittelt keinen allzu stabilen Eindruck, hält aber auch gröberen Genies stand. Die Bedienung erfolgt über drei Tasten und den Zeichenarm. Die Tasten öffnen den Weg ins Menü und zurück und ermöglichen außerdem die Bestätigung der Menüwahl, der Zeichenarm zeichnet und hilft beim Lesen der diversen Speisekarten. Den Computer selbst kann man bei der Arbeit getrost vergessen, alle Steuerbefehle erfolgen von der Tafel aus. Völlig Unbegabte legen sich jetzt eine Vorlage aufs Brett und ziehen mit dem Zeichenarm die Konturen nach – das Ergebnis zeigt sich auf

Scan: S. Höltgen


```
,246,102,126,124
10020 DATA24,36,90,153
,90,36,24,0
10030 DATA0,60,126,255
,255,94,44,20
10040 DATA255,129,189,
165,165,189,129,255
10050 DATA0,0,128,96,3
0,61,126,255
10060 DATA0,56,7,0,0,0
,0,0
10070 DATA0,0,0,0,0,19
2,60,0
10080 DATA0,0,1,6,120,
188,126,255
10090 DATA0,0,0,0,0,3,
60,0
10100 DATA0,60,192,0,0
,0,0,0
10110 DATA0,14,14,14,1
4,14,14,0
10120 DATA0,112,112,11
2,112,112,112,0
10130 DATA16,124,16,56
,68,196,60,0
10140 DATA0,0,0,0,0,0,
56,56
10150 DATA8,62,8,28,34
,35,60,0
10160 DATA0,0,0,0,0,0,
28,28
10170 DATA0,0,255,0,25
5,0,255,0
10180 DATA255,255,255,
255,255,255,255,255
10190 DATA65,0,73,42,2
8,42,129,65

READY.
```



Olympia

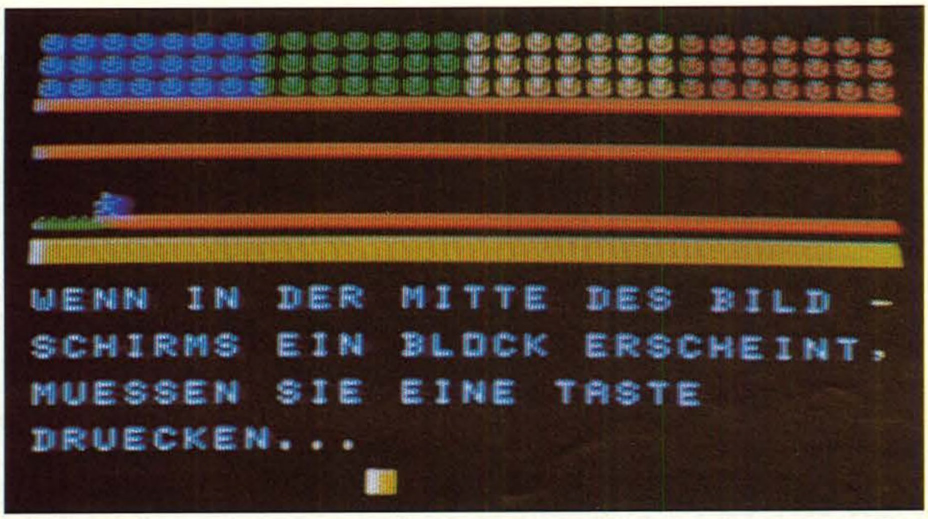
Als Mitglied der Nationalmannschaft sollen Sie mit dem TI-99/4A in der Grundversion in fünf leichtathletischen Disziplinen Ihr Reaktionsvermögen unter Beweis stellen

Der Wettkampf besteht aus den fünf Disziplinen
100-Meter-Lauf
Weitsprung
120 Meter Hürden
Kugelstoßen
Hindernislauf

Die wichtigsten Variablen im Programm sind:
ZEIT: Zeitzähler während der Reaktionsphase
PUNKTE: Gesamtpunktestand, der sich von Disziplin zu Disziplin verringert
METER: Meter-Berechner während des Weitsprungs oder des Kugelstoßens

Die Grafikzeichen sind unter folgenden Code-Zahlen abgelegt:
96: laufende Figur
97: laufende Figur
98: springende Figur
104: Grasboden
112: Sandboden
113: Block
120: Steinboden
103: Zuschauer
111: Zuschauer
119: Zuschauer
123: Zuschauer

Volker Niemeyer




```
10 REM  OLYMPIA
        (C) 1984
        TI 99/4A
20 REM  VOLKER NIEMEYER
        UESENER RING 30
        2807 ACHIM
30 REM  * EINLEITUNG *
40 CALL CLEAR
50 CALL SCREEN(2)
60 PRINT " .  .    . . .    .
.. .  .  . . .    . . .. ..
. . . . . . . .   . . .
. . .. . ."
70 PRINT ". . .      .  .    .
.   . . .  .    ..  .  .   .
.   . . ."::::::::::::::
"T A S T E   D R U E C K E N
"::
80 FOR C=1 TO 12
90 CALL COLOR(C,8,1)
95 NEXT C
100 CALL COLOR(2,11,2)
110 CALL KEY(0,K,S)
120 IF S=0 THEN 110
130 CALL CLEAR
140 REM  * FIGUREN DEFINIERE
N *
150 CALL CHAR(96,"303822FCB0
38E88C")
160 CALL CHAR(97,"181C907E1A
78480C")
170 CALL CHAR(98,"181C117E91
1A7E40")
180 CALL CHAR(103,"003C5A7E5
A663C00")
190 CALL COLOR(9,5,1)
200 CALL CHAR(104,"2449FFFF0
0000000")
210 CALL CHAR(111,"003C5A7E5
A663C00")
220 CALL COLOR(10,13,1)
230 CALL CHAR(112,"2277FFFF0
0000000")
240 CALL CHAR(113,"FFFFFFFFF
FFFFFFF")
250 CALL CHAR(119,"003C5A7E5
A663C00")
260 CALL COLOR(11,11,1)
270 CALL CHAR(120,"FFFFFFFF0
0000000")
280 CALL CHAR(123,"003C5A7E5
A663C00")
290 CALL COLOR(12,7,1)
300 PRINT "HALLO SPORTLER !!
!"::"SIE SIND DER FUENFKAEMP
FER"::"IHRER MANNSCHAFT.
FOLGENDE"::
310 PRINT "DISZIPLINEN MUESS
EN SIE"::"MEISTERN UM WELTME
ISTER ZU"::"WERDEN:"::"1
00 METER LAUF..."::
320 PRINT "WEITSPRUNG..."::"
120 METER HUERDEN..."::"KUGE
LSTOSSEN..."::"HINDERNIS
SLAUF..."::
330 PRINT "T A S T E   D R U
 E C K E N"
340 CALL KEY(0,K,S)
350 IF S=0 THEN 340
360 CALL CLEAR
370 REM  * VARIABLEN FESTLEG
EN *
380 PUNKTE=10000
390 REM  * SPIELBEGINN *
400 CALL CLEAR
410 PRINT "ggoooow{{{{{gow{{
g{owgwwwogowg{{ow{{gowow{{ow
g{{wo{{gwo{{ggwo{{gwowgo
w{{owg{{{oooooo"
420 PRINT "xxxxxxxxxxxxxxxxxx
xxxxxxxxxxx
           xxxxxxxxxxxxx
xxxxxxxxxxxxxxx":::
430 PRINT "hhhhhhhhhhhhhhhhh
hhhhhhhhhhqqqqqqqqqqqqqqqqq
qqqqqqqqqqq"::
440 PRINT ::::"V I E L   S P
 A S S ! ! !":::::::
450 FOR P=1 TO 1000
460 NEXT P
500 CALL CLEAR
510 PRINT "gggggggoooooooowww
wwww{{{{{{{gggggggoooooooowww
wwww{{{{{{{gggggggoooooo
owwwwwww{{{{{{{"
520 PRINT "xxxxxxxxxxxxxxxxx
xxxxxxxxxxx"::"xxxxxxxxxxxxx
xxxxxxxxxxxxxxx"::"   `":
"hhxxxxxxxxxxxxxxxxxxxxxxxxx
x"
530 PRINT "qqqqqqqqqqqqqqqqq
qqqqqqqqqqq"::"WENN IN DER M
ITTE DES BILD -"::"SCHIR
MS EIN BLOCK ERSCHEINT,"::
540 PRINT "MUESSEN SIE EINE
TASTE"::"DRUECKEN..."::::::
550 RANDOMIZE
560 C=INT(RND*300)+150
570 FOR A=1 TO C
575 CALL KEY(0,K,S)
576 IF K>31 THEN 582
580 NEXT A
581 GOTO 590
582 PRINT "FEHLSTART..."
584 FOR P=1 TO 1000
586 NEXT P
588 GOTO 500
590 CALL HCHAR(20,14,113)
600 CALL KEY(0,K,S)
605 LET ZEIT=ZEIT+1
610 IF S=0 THEN 590
620 GOTO 700
700 FOR B=5 TO 31
705 LET ZEIT=ZEIT+1
710 CALL SOUND(-100,-8,2)
720 CALL SOUND(-100,-4,2)
730 IF A=2 THEN 770
740 CALL HCHAR(8,B,97)
750 CALL HCHAR(8,B-1,32)
755 A=2
760 GOTO 800
770 CALL HCHAR(8,B,96)
780 CALL HCHAR(8,B-1,32)
790 A=1
800 NEXT B
805 PRINT "100 METER LAUF :
ERGEBNIS..."::
810 PRINT "ZEIT = ";ZEIT
820 PUNKTE=PUNKTE-(ZEIT*10)
830 FOR P=1 TO 1000
840 NEXT P
900 CALL CLEAR
910 PRINT "oooooooooooooooooo
oooooooooooggggggggggggggggg
ggggggggggggwwwwwwwwwwwwww
w{{{{{{{{{{{{{{{"
920 PRINT "xxxxxxxxxxxxxxxxx
xxxxxxxxxxx"::"xxxxxxxxxxxxx
xxxxxxxxxxxxxxx"::"    `"
930 PRINT "hhhxxxxxxxxxxxpppp
ppppppppppqqqqqqqqqqqqqqqqq
qqqqqqqqqqq"::
940 PRINT "WENN EIN BLOCK ER
SCHEINT,"::"TASTE DRUECKEN":
::::
950 LET METER=14
955 ZEIT=0
960 RANDOMIZE
970 C=INT(RND*300)+150
980 FOR A=1 TO C
990 CALL KEY(0,K,S)
1000 IF K>31 THEN 1020
1010 NEXT A
1015 GOTO 1060
1020 PRINT "FEHLSTART..."
1030 FOR P=1 TO 1000
1040 NEXT P
1050 GOTO 900
1060 CALL HCHAR(21,14,113)
1070 CALL KEY(0,K,S)
1080 LET ZEIT=ZEIT+1
1090 IF S=0 THEN 1070
1100 FOR B=6 TO 16
1105 CALL SOUND(-100,-8,2)
1106 CALL SOUND(-100,-4,2)
1110 IF A=2 THEN 1160
1120 CALL HCHAR(13,B,96)
1130 CALL HCHAR(13,B-1,32)
1140 A=2
1150 GOTO 1190
1160 CALL HCHAR(13,B,97)
1170 CALL HCHAR(13,B-1,32)
1180 A=1
1190 NEXT B
1200 LET METER=METER-ZEIT
1210 IF METER<1 THEN 1020
1220 CALL HCHAR(13,B-1,32)
1230 FOR B=17 TO (17+METER)
1240 CALL SOUND(-100,1660,4)
1250 CALL HCHAR(12,B,98)
1260 CALL HCHAR(12,B-1,2)
1270 NEXT B
1280 CALL HCHAR(12,B-1,32)
1290 CALL HCHAR(13,B-1,96)
1300 PRINT "WEITSPRUNG..."::
"METER = ";METER
1310 PUNKTE=PUNKTE-(ZEIT*100
)
1320 FOR P=1 TO 1000
1330 NEXT P
```

```
1400 CALL CLEAR
1410 PRINT "gow{gow{gow{gow{
gow{gow{gow{ggow{gow{gow{gow
{gow{gow{gowoggow{gow{go
w{gow{gow{gow{go"
1420 PRINT "xxxxxxxxxxxxxxxx
xxxxxxxxxxxx"::"xxxxxxxxxxxx
xxxxxxxxxxxxxxxx"::"`
#    #    #    #    #"
1430 PRINT "xxxxxxxxxxxxxxxx
xxxxxxxxxxxxqqqqqqqqqqqqqqqq
qqqqqqqqqqq"::
1440 PRINT "WENN BLOCK ERSCH
EINT,TASTE"::"DRUECKEN,UND M
IT TASTE G"::"UEBER DIE
HINDERNISSE"::"SPRINGEN...":
:::
1450 RANDOMIZE
1460 C=INT(RND*300)+150
1470 FOR A=1 TO C
1480 CALL KEY(0,K,S)
1490 IF K>31 THEN 1510
1500 NEXT A
1505 GOTO 1540
1510 PRINT "FEHLSTART..."
1520 FOR P=1 TO 1000
1530 NEXT P
1535 GOTO 1400
1540 B=3
1541 CALL HCHAR(22,14,113)
1542 CALL KEY(0,K,S)
1543 ZEIT=ZEIT+1
1544 IF S=0 THEN 1542
1550 LET B=B+1
1551 IF B=7 THEN 1770
1552 IF B=12 THEN 1770
1553 IF B=17 THEN 1770
1554 IF B=22 THEN 1770
1555 IF B=27 THEN 1770
1556 CALL SOUND(-100,-4,2)
1557 CALL SOUND(-100,-8,2)
1560 IF A=2 THEN 1610
1570 CALL HCHAR(10,B,96)
1580 CALL HCHAR(10,B-1,32)
1590 A=2
1600 GOTO 1650
1610 CALL HCHAR(10,B,97)
1620 CALL HCHAR(10,B-1,32)
1630 A=1
1650 CALL KEY(0,K,S)
1660 IF K=71 THEN 1680
1665 IF B>29 THEN 1800
1670 GOTO 1550
1680 CALL HCHAR(10,B,32)
1690 CALL HCHAR(9,B+1,98)
1700 CALL SOUND(-100,1660,4)
1710 CALL HCHAR(9,B+1,32)
1720 B=B+2
1730 CALL HCHAR(10,B,96)
1740 LET ZEIT=ZEIT+4
1750 IF B>29 THEN 1800
1760 GOTO 1550
1770 LET ZEIT=ZEIT+6
1780 CALL SOUND(-100,110,2)
1790 GOTO 1556
1800 PRINT "ERGEBNIS 120 MET
ER HUERDENL."::"ZEIT : ";ZEI
T
1810 PUNKTE=PUNKTE-(ZEIT*10)
1820 FOR P=1 TO 1000
1830 NEXT P
1900 CALL CLEAR
1901 ZEIT=0
1910 PRINT "w    o  ggg     {
w  g o {{ {     {  ww o gg o
 ww  g o  ww{{{{{wwwwogo
gogwgowogo{{{ow{"
1920 PRINT "xxxxxxxxxxxxxxxx
xxxxxxxxxxxx"::"xxxxxxxxxxxx
xxxxxxxxxxxxxxxx"::" `":
"xxxhhhhhhhhhhhhhhhhhhhhhhhhh
h"::
1930 PRINT "qqqqqqqqqqqqqqqq
qqqqqqqqqqq"::"WENN BLOCK E
RSCHEINT,TASTE"::"DRUECK
EN..."::::::
1940 RANDOMIZE
1950 C=INT(RND*300)+150
1960 FOR A=1 TO C
1970 CALL KEY(0,K,S)
1980 IF K>31 THEN 2000
1990 NEXT A
1999 GOTO 2040
2000 PRINT "FEHLSTART..."
2010 FOR P=1 TO 1000
2020 NEXT P
2030 GOTO 1900
2040 CALL HCHAR(20,14,113)
2050 CALL KEY(0,K,S)
2060 LET ZEIT=ZEIT+1
2070 IF S=0 THEN 2050
2080 METER=25
2085 LET METER=METER-ZEIT
2086 IF METER<1 THEN 2200
2090 FOR B=5 TO (METER+5)
2100 CALL HCHAR(10,B,46)
2110 CALL HCHAR(10,B-1,32)
2120 CALL SOUND(-100,880,4)
2130 NEXT B
2140 CALL HCHAR(10,B-1,32)
2150 CALL HCHAR(11,B-1,46)
2160 PRINT "ERGEBNIS KUGELST
OSSEN..."::"METER = ";METER
2170 PUNKTE=PUNKTE-(ZEIT*100
)
2180 GOTO 2240
2200 PRINT "FEHLSTART..."
2210 FOR P=1 TO 1000
2220 NEXT P
2230 GOTO 1900
2240 FOR P=1 TO 1000
2250 NEXT P
2300 CALL CLEAR
2310 PRINT " g  o  w w {    {
 gg  o  w   g g o  w w {{ {{
 g g o w w  g g o  w w {
 { { gg  o  w    "
2320 PRINT "g g o   w  {    {
 g   o w w   g   oo w  {    {
 g   o w w  xxxxxxxxxxxx
xxxxxxxxxxxxxxxx"
2330 PRINT :"xxxxxxxxxxxxxxx
xxxxxxxxxxxx"::" `        #
   @     #       xxxxxxx
xxxx xxxxxxxxxxxxxxxx"
2340 PRINT "qqqqqqqqqqqqqqqq
qqqqqqqqqqq"::"WENN EIN BLO
CK ERSCHEINT,"::"TASTE D
RUECKEN UND MIT G"::
2350 PRINT "UEBER DIE HINDER
NISSE"::"SPRINGEN..."::::
2355 ZEIT=0
2360 RANDOMIZE
2370 C=INT(RND*300)+150
2380 FOR A=1 TO C
2390 CALL KEY(0,K,S)
2400 IF K>31 THEN 2430
2410 NEXT A
2420 GOTO 2470
2430 PRINT "FEHLSTART..."
2440 FOR P=1 TO 1000
2450 NEXT P
2460 GOTO 2300
2470 CALL HCHAR(23,14,113)
2480 CALL KEY(0,K,S)
2490 LET ZEIT=ZEIT+1
2500 IF S=0 THEN 2480
2510 B=4
2520 B=B+1
2522 CALL SOUND(-100,-8,2)
2524 CALL SOUND(-100,-4,2)
2525 IF B=11 THEN 2700
2526 IF B=14 THEN 2700
2527 IF B=17 THEN 2700
2528 IF B=23 THEN 2700
2529 IF A=2 THEN 2580
2530 CALL HCHAR(11,B,96)
2540 CALL HCHAR(11,B-1,32)
2550 A=2
2560 GOTO 2620
2580 CALL HCHAR(11,B,97)
2590 CALL HCHAR(11,B-1,32)
2600 A=1
2620 CALL KEY(0,K,S)
2630 IF K=71 THEN 2660
2640 LET ZEIT=ZEIT+1
2645 IF B>29 THEN 2740
2650 GOTO 2520
2660 CALL HCHAR(11,B,32)
2665 CALL HCHAR(10,B+1,98)
2670 CALL HCHAR(10,B+1,32)
2675 CALL SOUND(-100,1660,2)
2680 CALL HCHAR(11,B+2,96)
2685 B=B+2
2690 ZEIT=ZEIT+4
2692 IF B>29 THEN 2740
2695 GOTO 2520
2700 CALL SOUND(-100,110,2)
2720 ZEIT=ZEIT+6
2730 GOTO 2529
2740 PRINT "HINDERNISSLAUF E
RGEBNIS..."::"ZEIT = ";ZEIT
2750 PUNKTE=PUNKTE-(ZEIT*10)
2760 FOR P=1 TO 1000
2770 NEXT P
2800 CALL CLEAR
2810 IF PUNKTE>5500 THEN 290
0
2820 PRINT "DAS WAR EINE SOL
CHE"::"SCHLECHTE LEISTUNG VO
N IHNEN"::",DASS SIE IN
DIE BEZIRKSLIGA"::
2830 PRINT "ZURUECKVERSETZT
WERDEN..."::"PUNKTE = ";PUNK
TE:::::::
2890 GOTO 2890
2900 IF PUNKTE>7000 THEN 300
0
2910 PRINT "WEGEN IHRER SCHL
ECHTEN"::"LEISTUNG BEI DIESE
R WELT -"::"MEISTERSCHAF
T SCHEIDEN SIE"::
2920 PRINT "AUS DER NATIONAL
MANNSCHAFT"::"IHRES LANDES A
US..."::"PUNKTE = ";PUNK
TE::::::
2930 GOTO 2930
3000 IF PUNKTE>7700 THEN 310
0
3010 PRINT "IHR ERGEBNIS WAR
 MITTEL -"::"MAESSIG.SIE DUE
RFEN TROTZ -"::"DEM NOCH
 IN DER NATIONAL -"::
3020 PRINT "BLEIBEN..."::"PU
NKT = ";PUNKTE:::::::
3100 IF PUNKTE>8000 THEN 325
0
3110 PRINT "IHR ERGEBNIS WAR
 GANZ GUT."::"SIE BLEIBEN IN
 DEM TEAM..."::"PUNKTE =
 ";PUNKTE::
3120 IF PUNKTE>7750 THEN 315
0
3130 PRINT "SIE ERREICHTEN D
EN 3.PLATZ"::"GRATULIERE..."
::
3140 GOTO 3140
3150 PRINT "SIE ERREICHTEN D
EN 2.PLATZ"::"GRATULIERE..."
::
3160 GOTO 3160
3250 PRINT "                qqqqq
                        q  qq
                        q
 q q"
3260 PRINT "            q  qq
             PUNKTE     q qqq
   ";PUNKTE;"
q qqq"
3270 PRINT "            qqqqq
                         qqq

 q
 q"
3280 PRINT "             qqq
                        qqqqq
"::"F A N T A S T I S C
H ! ! !"::
3290 PRINT "S I E   S I N D"
::"W E L T M E I S T E R"::"
I M   F U E N F K A M P
F !"::"G R A T U L I E R E !
!!":
3300 GOTO 3300
```

Scan: S. Höltgen

Eine Blume geht ein, wenn sie zuviel gegossen wird. Bei zuwenig Gießen hat sie schließlich gar keine Blätter mehr und stirbt ebenfalls ab.

Die gespielten Sekunden zählen Punkte. Das Spiel hat eine High-Score-Wertung. Gespielt wird mit Joystick; gegossen wird aus einer Wolke mit der Feuertaste.

Nach dem Start erscheint auf dem Bildschirm das Copyright. In zirka zwei Minuten wird der Zeichensatz kopiert und geändert (Zeilen 280–430) sowie das Maschinenprogramm (Zeilen 12000–12150 mit den DATA-Zeilen 18001–18061) eingepoket.

Bedienung

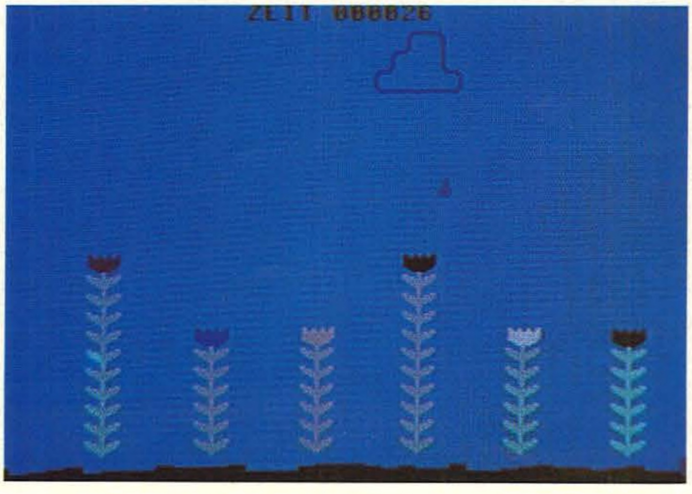
Wird die Aufforderung, den Feuerknopf zu drücken, nicht befolgt, folgt nach einer Warteschleife (Zeilen 10140–10165) von zirka 20 Sekunden die Spielanleitung. Wird der Feuerknopf betätigt, können im nachfolgenden Teil (Zeilen 470–620) auf dem Bildschirm mit dem Joystick drei Buchstaben als Abkürzung für den Namen gewählt werden.

Ein weiteres Rechtsdrücken des Joysticks bewirkt das Erscheinen der Blumen (Zeilen 690–970). In Zeile 1010 wird TI$ gesetzt und in Zeile 1020 geprintet.

Das eigentliche Spiel läuft in den Programmzeilen 1000–1090 zusammen mit den drei Unter-Programmen (Zeilen 2000, 3000, 4000) ab. Nach Ende des Spiels erfolgen eine Auswertung und eine Anzeige auf dem Bildschirm (Zeilen 6900–7150).

Blumen

Ein bezauberndes Denk- und Geschicklichkeitsspiel für den C 64, bei dem Pflanzen durch wohldosiertes Gießen möglichst lange Zeit erhalten werden sollen



Korrekte Eingabe

Die korrekte Eingabe der DATA-Zeilen wird über Check-Summen (Zeilen 230–250) kontrolliert, und eventuelle Eingabefehler werden angezeigt (Zeilen 13000 bis 13040). In den Zeilen 50–160 werden die Daten für die Sprites übergeben. Die Bewegung der Sprites ermöglicht das Unterprogramm ab Zeile 19000. Wir wünschen viel Spaß.

T. Bagdonat und *Dr. Siegfried Bagdonat*

```
4040 ifb<=vthen4020
4050 n=(b-v)/z:goto4090
4060 ifb$<>"-"then4080
4070 b=v+z*n:goto4090
4080 v=b-z*n
4070 print"■Ermittelte Werte:":print"■"v;b;n
4100 print"■Wollen Sie die Eingabe wiederholen ?":poke198,0
4110 geta$:ifa$="j"then4000
4120 ifa$<>"n"then4110
4130 return
5000 open1,8,2,"0:za.p":open2,8,15:input#2,a:ifa<>0thenclose1:close2:run
5010 input#1,xv,xb,yv,yb,sx,sy,rx,ry:close1:close2:
5015 print"■":open1,4,10:print#1,:close1:open1,4:print#1,chr$(27)+"2"
5020 pokefa,af:sysg,p:print#1,:print#1,"X-Achse:"
5030 print#1,"Von:";xv;", bis:";xb:print#1,"Teilstrichabstand:";rx
5040 print#1,"Punktabstand:";sx:print#1,:print#1,"Y-Achse:"
5050 print#1,"Von:";yv;", bis:";yb:print#1,"Teilstrichabstand:";ry
5060 print#1,"Punktabstand:";sy:print#1,chr$(7):poke198,0:pokefa,wf
5070 print"Papierauswurf (J/N) ?"
5080 geta$:ifa$="n"thenclose1:goto80
5090 ifa$<>"j"then5080
5100 print#1,chr$(12)+chr$(7);:close1:goto80
6000 open1,8,2,"0:za.p":open2,8,15:input#2,a:ifa<>0thenclose1:close2:run
6010 input#1,xv,xb,yv,yb,sx,sy,rx,ry:close1:close2:goto460
ready.
```

Shooting

Nicht mit Kanonen auf Spatzen, sondern mit einem Raumschiff auf giftige Pilze wird bei diesem Programm für den TI-99/4A in der Grundversion geschossen

Das Raumschiff dreht sich nach einem Druck auf die Taste „1“ um 90 Grad im Uhrzeigersinn. Wird diese Taste losgelassen fährt das Raumschiff in die Richtung, in die die Spitze zeigt. Außer den Sternen stehen im Feld verstreut Sterne. Stößt das Schiff gegen einen Stern, Pilz oder die Umrandung, explodiert es.

Pilz im Visier

Die verbliebenen Schiffe werden im rechten unteren Eck angezeigt. Sind alle Schiffe verloren, ist das Spiel aus.

Um die Pilze zu beseitigen, muß auf die Leertaste gedrückt werden. Danach löst sich vom Schiff ein Schuß. Dieser fliegt in Richtung der Spitze des Schiffes. In dieser Zeit bleibt das Schiff stehen.

Trifft der Schuß einen Pilz, explodiert dieser, und der Spieler bekommt 100 Punkte auf sein Punktekonto gutgeschrieben.

Trifft er jedoch auf einen Stern oder die Bildschirmumrandung, wird Treibstoff aus dem Tank des Spielers abgezogen. Das gleiche gilt, wenn das Schiff gedreht wird.

Ist der Tank leer, geht ein Schiff verloren. Wird jedoch ein Pilz abgeschossen, bekommt der Spieler Treibstoff gutgeschrieben. Der Tank wird als weißer Strich in der unteren Bildschirmhälfte dargestellt.

Ein Freischiff zur Belohnung

Alle 1500 Punkte wird der Spieler gelobt und der Schwierigkeitsgrad gesteigert. Bei 4500 Punkten bekommt man ein Freischiff. Bleibt noch zu erwähnen, daß am Anfang des Spiels der Schwierigkeitsgrad eingestellt werden kann. Der Schwierigkeitsgrad ist zwischen 1 (der schwerste) und 4 (der leichteste) wählbar.

Eine kleine Verschnaufpause

Auch kann das Spiel mit der Taste „P“ unterbrochen werden. Auf dem Bildschirm erscheint dann die Meldung „PAUSE“, und das Spiel steht still. Durch nochmaliges Drükken der Taste „P“ wird das Spiel fortgesetzt.

Und jetzt viel Spaß beim Spielen! *Michael Schneller*

Scan: S. Höltgen

Scan: S. Höltgen


```
5 REM   ******************
7 REM
10 REM   ** SHOOTING **
12 REM
14 REM  * FUER TI 99/4A *
16 REM
18 REM  ** IN TI-BASIC **
20 REM
30 REM   * COPYRIGHT BY *
40 REM
50 REM  MICHAEL SCHNELLER
60 REM
70 REM     *** 1983 ***
75 REM
80 REM  ******************
85 CALL CLEAR
86 RESTORE 91
91 DATA 66,89,32,77,73,67,72
,65,69,76,32,83,67,72,78,69,
76,76,69,82
92 DATA 66,69,84,82,85,71
93 FOR R=1 TO 20
94 READ XY
95 Z=Z+XY
96 NEXT R
97 IF Z<>1390 THEN 2600
99 RESTORE 91
100 CALL CLEAR
110 RANDOMIZE
120 SHIP=0
130 POINT=0
135 LEVEL=1
137 LS=2
140 CT=0
150 CALL CHAR(96,"3C3C7EFFFF
3C3C3C")
151 CALL CHAR(95,"FFFFFFFFFF
FFFFFF")
160 CALL CHAR(104,"FFFFFFFFF
FFFFFFF")
170 CALL CHAR(105,"995A3CFFF
F3C5A99")
180 CALL CHAR(112,"101038387
C7CFEFE")
190 CALL CHAR(113,"C0F0FCFFF
CF0C")
200 CALL CHAR(114,"7F7F3E3E1
C1C0808")
210 CALL CHAR(115,"00030F3FF
F3F0F03")
220 CALL CHAR(128,"000000181
8")
230 CALL CHAR(129,"000024000
024")
240 CALL CHAR(130,"004200000
00042")
250 CALL CHAR(131,"810000000
0000081")
260 CALL CHAR(132,"FFFFFFFFF
FFFFFFF")
270 CALL CHAR(136,"887548972
5473416")
280 CALL CHAR(137,"485475634
5388146")
290 CALL CHAR(138,"587364543
6574165")
300 GOSUB 2320
310 DIM CH(4)
320 IX=112
330 FOR I=1 TO 4
340 CH(I)=IX
350 IX=IX+1
360 NEXT I
370 DIM XR(4)
380 DIM YR(4)
390 XR(1)=0
400 YR(1)=-1
410 XR(2)=1
420 YR(2)=0
430 XR(3)=0
440 YR(3)=1
450 XR(4)=-1
460 YR(4)=0
470 CALL COLOR(9,4,1)
480 FOR C=2 TO 8
490 CALL COLOR(C,12,1)
500 NEXT C
510 CALL COLOR(13,16,1)
520 CALL SCREEN(2)
530 CALL COLOR(10,5,1)
540 CALL COLOR(11,16,1)
550 LH=18
560 CALL HCHAR(24,7,132,LH)
570 CALL COLOR(14,16,1)
580 CALL COLOR(15,16,1)
590 CALL COLOR(16,16,1)
600 CALL HCHAR(2,1,104,32)
610 CALL VCHAR(2,1,104,20)
620 CALL VCHAR(2,32,104,20)
625 CALL HCHAR(22,28,112,LS)
630 CALL HCHAR(21,1,104,32)
640 FOR I=1 TO ANZ
650 XH=INT(30*RND)+2
660 YH=INT(18*RND)+3
662 CALL GCHAR(YH,XH,A)
663 IF ANZ>350 THEN 670
664 IF A<>32 THEN 650
670 CALL HCHAR(YH,XH,105)
680 NEXT I
690 FOR I=1 TO 15
700 XA=INT(30*RND)+2
710 YA=INT(18*RND)+3
720 CALL GCHAR(YA,XA,Q)
730 IF Q=96 THEN 700
740 CALL HCHAR(YA,XA,96)
750 NEXT I
760 CALL CHAR(116,"000000181
8")
770 P$="PUNKTE:"
780 X=4
790 Y=1
800 GOSUB 2460
801 P$="RUNDE="
802 X=20
803 Y=1
804 GOSUB 2460
805 P$=STR$(LEVEL)
806 X=26
807 Y=1
808 GOSUB 2460
810 P$=STR$(POINT)
820 Y=1
830 X=11
840 GOSUB 2460
841 P$="SCHWIERIGKEITSGRAD:"
842 X=2
843 Y=22
844 GOSUB 2460
845 P$=STR$(SG)
846 Y=22
847 X=21
848 GOSUB 2460
850 P$="TANK:"
860 X=2
870 Y=24
880 GOSUB 2460
890 IND=1
900 XS=INT(30*RND)+2
910 YS=INT(18*RND)+3
920 CALL GCHAR(YS,XS,A)
930 IF A<>32 THEN 900
940 CALL HCHAR(YS,XS,CH(IND)
)
950 FOR BG=1 TO 7
960 CALL HCHAR(YS,XS,32)
970 CALL SOUND(200,110,6)
980 CALL HCHAR(YS,XS,CH(IND)
)
990 CALL SOUND(200,330,6)
1000 NEXT BG
1010 GOSUB 1070
1020 GOTO 1010
1030 REM
1040 REM
1050 REM
1060 REM
1070 CALL KEY(0,K,S)
1080 IF K=49 THEN 1190
1090 IF K=32 THEN 1430
1095 IF K=80 THEN 3000
1100 CALL GCHAR(YS+YR(IND),X
S+XR(IND),V)
1110 IF V=32 THEN 1130
1120 GOTO 1300
1130 CALL HCHAR(YS,XS,32)
1140 XS=XS+XR(IND)
1150 YS=YS+YR(IND)
1160 CALL SOUND(-300,-5.64,6
)
1170 CALL HCHAR(YS,XS,CH(IND
))
1180 RETURN
1190 IND=IND+1
1200 IF IND>4 THEN 1280
1210 CALL SOUND(-300,-5.67,6
)
```

Scan: S. Höltgen

```
1220 CALL HCHAR(YS,XS,CH(IND
))
1230 LH=LH-1
1240 IF LH<1 THEN 1300
1250 CALL HCHAR(24,7,32,LH+2
)
1260 CALL HCHAR(24,7,132,LH)
1270 RETURN
1280 IND=1
1290 GOTO 1210
1300 REM  explosion des
1310 REM  schiffes!!
1320 REM  ***************
1330 CALL SOUND(2000,-7,2)
1340 FOR AEX=1 TO 10
1350 FOR INEX=136 TO 138
1360 CALL HCHAR(YS,XS,INEX)
1370 NEXT INEX
1380 NEXT AEX
1390 CALL CLEAR
1400 SHIP=SHIP+1
1405 LS=LS-1
1410 IF SHIP=3 THEN 2080
1420 GOTO 310
1430 XPS=XS+XR(IND)
1435 LAUT=0
1440 YPS=YS+YR(IND)
1450 CALL GCHAR(YPS,XPS,STEI
N)
1460 IF STEIN=105 THEN 1600
1470 IF STEIN=104 THEN 1690
1480 IF STEIN=96 THEN 1770
1490 CALL HCHAR(YPS,XPS,116)
1500 REM
1510 CALL HCHAR(YPS,XPS,32)
1520 XPS=XPS+XR(IND)
1530 YPS=YPS+YR(IND)
1535 CALL SOUND(-500,200,LAU
T)
1536 LAUT=LAUT+1
1540 CALL GCHAR(YPS,XPS,SC)
1550 IF SC=104 THEN 1690
1560 IF SC=105 THEN 1600
1570 IF SC=96 THEN 1770
1580 CALL HCHAR(YPS,XPS,116)
1590 GOTO 1500
1600 CALL HCHAR(YPS,XPS,32)
1610 FOR I=1 TO 30 STEP 3
1620 CALL SOUND(-100,-5.91,I
)
1630 NEXT I
1640 CALL SOUND(500,-7.34,5)
1650 FOR EX=128 TO 131
1660 CALL HCHAR(YPS,XPS,EX)
1670 NEXT EX
1680 CALL HCHAR(YPS,XPS,32)
1690 CALL HCHAR(24,7,32,LH)
1700 LH=LH-2
1710 IF LH<1 THEN 1300
1720 CALL HCHAR(24,7,132,LH)
1730 REM
1740 REM
1750 REM
1760 RETURN
1770 REM
1780 FOR SOUND=1 TO 30 STEP
3
1790 CALL SOUND(-34,-1.45,SO
UND)
1800 NEXT SOUND
1810 CALL SOUND(-600,110,10,
-7.23,5)
1820 FOR EX=140 TO 153
1830 CALL HCHAR(YPS,XPS,EX)
1840 NEXT EX
1850 CALL HCHAR(YPS,XPS,32)
1860 POINT=POINT+100
1870 X=11
1880 Y=1
1890 P$=STR$(POINT)
1900 GOSUB 2460
1910 CT=CT+1
1920 IF CT=15 THEN 1970
1930 IF LH+SG>25 THEN 1960
1940 LH=LH+SG
1950 CALL HCHAR(24,7,132,LH)
1960 RETURN
1970 CALL CLEAR
1980 P$="GAR NICHT SCHLECHT!
!"
1982 LEVEL=LEVEL+1
1983 IF LEVEL=4 THEN 2800
1990 X=6
2000 Y=12
2010 GOSUB 2460
2020 ANZ=ANZ+20
2030 FOR D=1 TO 100
2040 NEXT D
2050 CALL CLEAR
2060 CT=0
2070 GOTO 310
2080 RESTORE 91
2081 FOR I=1 TO 20
2082 READ ZX
2083 H$=H$&CHR$(ZX)
2084 NEXT I
2085 PRF$=H$
2086 Z=0
2089 PRINT " ERREICHTE PUNKT
E=";POINT
2090 PRINT
2100 PRINT "hhhhhhhhhhhhhhhh
hhhhhhhhhhhh"
2110 PRINT
2120 PRINT "‘ i  i i  hh ‘i‘
 p r q ‘"
2130 PRINT "q ii  ‘p ‘ i h q
   s h  i"
2140 PRIN
2150 PRINT
2160 PRINT
2170 PRINT ""
2180 PRINT
2190 PRINT "      S H O O T I
 N G"
2200 PRINT
2210 PRINT ""
2220 PRINT
2230 PRINT "      COPYRIGHT 1
983"
2240 PRINT
2249 IF H$<>PRF$ THEN 2600
2250 PRINT "  ";H$
2255 H$=""
2260 PRINT
2270 PRINT
2280 PRINT " PRESS ANY KEY T
O BEGIN"
2290 CALL KEY(0,K,S)
2300 IF S<>0 THEN 10
2310 GOTO 2290
2320 INPUT "SCHWIERIGKEITSGR
AD?(1-4):":SG
2330 CALL CLEAR
2340 IF SG<1 THEN 2320
2350 IF SG>4 THEN 2320
2360 ON SG GOTO 2380,2400,24
20,2440
2370 CALL CLEAR
2380 ANZ=100
2390 RETURN
2400 ANZ=80
2410 RETURN
2420 ANZ=60
2430 RETURN
2440 ANZ=40
2450 RETURN
2460 FOR STRA=1 TO LEN(P$)
2470 CALL HCHAR(Y,(X+STRA)-1
,ASC(SEG$(P$,STRA,1)))
2480 NEXT STRA
2490 CALL SOUND(160,1405,5)
2500 RETURN
2600 RESTORE 92
2610 FOR STP=1 TO 6
2620 READ ZAHL
2630 B$=B$&CHR$(ZAHL)
2640 NEXT STP
2650 PRINT TAB(12);B$
2660 FOR P=1 TO 24
2670 PRINT
2680 NEXT P
2690 END
2800 LS=LS+1
2810 CALL HCHAR(22,28,112,LS
)
2815 SHIP=SHIP-1
2820 GOTO 1990
3000 P$="PAUSE"
3010 X=14
3020 Y=21
3030 GOSUB 2460
3040 CALL KEY(0,K,S)
3050 IF K=80 THEN 3100
3060 GOTO 3040
3100 CALL HCHAR(21,1,104,32)
3110 GOTO 1070
```

Scan: S. Höltgen

Moskitojäger

Fünf Frösche sind auf Insektenfang. Gefräßige Ungeheuer im See warten nur auf eine ungeschickte Bewegung (TI-99/4A mit Extended-BASIC und Joystick)

In der ersten Phase des Spieles muß der Spieler versuchen, mit dem Frosch zum Insekt zu springen. Der Absprung des Frosches kann hier mit dem Feuerknopf ausgelöst werden.

Wurde das Insekt verfehlt, so stürzt der Frosch in den See, wo gefräßige Ungeheuer ihn schon begierig erwarten. Er kann sich nur retten, wenn er eine Leiter erreicht.

Bevor das Spiel in dieser Phase beginnt, ertönt ein hoher Ton. Durch Druck auf den Feuerknopf schaltet man diesen aus und kann den Frosch nun unter Wasser in allen Richtungen bewegen.

Rettende Leiter

Berühren des Grundes, der Wasseroberfläche oder eines Ungeheuers hat den sofortigen Tod zur Folge. Hat man die Leiter erreicht, springt der Frosch von selbst an Land. Insgesamt hat man fünf Frösche zur Verfügung.

Das Programm belegt circa sieben KByte-Speicher. Es stehen also noch sieben KByte für Verbesserungen und Erweiterungen zur Verfügung. So könnte man das Spiel erschweren, indem nur drei Frösche zur Verfügung stehen (wird in Zeile 10 festgelegt) oder indem die Geschwindig-

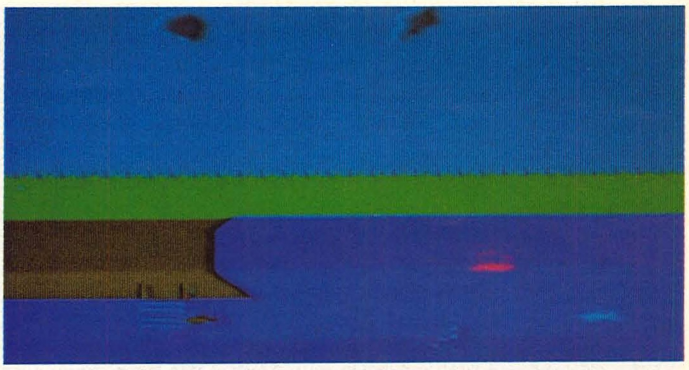
keit der Ungeheuer mit Verspeisen von nur einem Insekt erhöht wird (Zeile 590, nach jedem zehnten gefressenen Moskito wird die Geschwindigkeit um eine Einheit erhöht) oder indem nach jedem zehnten Moskito die Geschwindigkeit um zwei oder fünf Einheiten erhöht wird.

Anregungen zur Erweiterung

Weiterhin könnte man verschieden geformte Moskitos verwenden, eine Eingabe einfügen, bei der man zu Beginn des Spieles den Schwierigkeitsgrad festlegt (der Wert von Q bekommt einen Wert ungleich Null). Sicherlich würde „Moskitojäger" auch noch interessanter, wenn es für zwei oder mehr Spieler gleichzeitig zu spielen wäre, so daß man Wettbewerbe austragen kann.

Um das Spiel per Tastatur steuern zu können, müssen nur die Zeilen 640, 790 und 800 geändert werden. In Zeile 800 ist darauf zu achten, daß der CALL-MOTION-Befehl entsprechend angepaßt wird.

Obwohl das Spiel grafisch schon ziemlich ausgefeilt ist, bieten sich jedoch immer noch zahllose Verbesserungsmöglichkeiten. Die vielen Bemerkungen im Programm dürften die Arbeit dabei sehr erleichtern. *Lutz Stradmann*



```
10 FROSCH=5 :: MOSKITO=0 :: GEFRESSEN=0
:: Q=0
20 CALL CLEAR :: DISPLAY AT(12,8):"MOSKI
TOJAEGER" :: FOR D=1 TO 15 :: CALL SOUND
(80,450,0,430,0,410,0):: CALL SOUND(80,5
80,0,670,0,760,0):: NEXT D
30 REM ***********************
40 REM   PROGRAMMIERT VON
50 REM   LUTZ STRADMANN
60 REM   AUGUST 1983
70 REM   ALLE RECHTE
80 REM   VORBEHALTEN
90 REM ***********************
100 CALL MAGNIFY(3)
110 CALL CLEAR
120 RANDOMIZE
130 CALL CHAR(33,"001FFFFFFFFFFFFF000000
030F3F7FFF03070F1F3F7FFFFFFFFF7F3F1F0F07
03")!FLUSSKRUEMMUNG
140 CALL CHAR(37,"FF7F3F1F0F070300")!FLU
SSKRUEMMUNG
150 CALL CHAR(38,"FFFFFFFFFFFFFFFF")!SEE
160 CALL CHAR(52,"000008908552649A081224
281008449224140C0432090701928C4932FCC0B0
80")!WASSERPFLANZE
170 CALL CHAR(120,"0180432212720A091723C
3010304030720A0A0212E5090A010DEC080C080C
0E0")!ANEMONE
180 CALL CHAR(124,"033F291D2B756B765077F
AAEE6D7B4CA8080B0B8A8F4A47454B8A0C040789
457")!FELS
190 CALL CHAR(68,"0000030F1F3F3F1F1F1F7F
7F7FE860F0FCFCF8F8FCFEFEFEF8C0")!WOLKE 1
200 CALL CHAR(72,"00031F3F7F7F7FFFFF3F0F
07000000000F0F8FEFEFFFFFFFFFFFEFEFE7C3800
")!WOLKE
210 CALL CHAR(80,"08181D1D090F3FFF0C120A
000000000040C0C0808080F8F8609050")!INSEK
T
220 CALL CHAR(88,"0000000000000000000000
0FFF1F070100000000000000002161E3E3FFFFC3
01")!HAI
230 CALL CHAR(92,"0000000000001028484681
000000000000307041823477880808946303")
!SEESCHLANGE
240 CALL CHAR(96,"000008847209077F8F1721
22440800000000010204790E0F0FFE09442210200
00")!SEEIGEL
250 CALL CHAR(100,"000F5FFFFF5F0F0000000
0000000000063930A84840A9363000")!FROSCH
IM WASSER
260 CALL CHAR(104,"0000000000007083060400
0000001030010 2C2E7E78FF3820202020408 0800
000")!FLIEGENDER FROSCH
270 CALL CHAR(108,"1C18303018180C0D07070
7071B21636138180C0C181830B0E0E0E0E0D884C
686")!ABSTURZFROSCH
280 CALL CHAR(112,"000000000000031F3F7C7
36F7C18000000000000003CF6FCF8C000C0E070"
)!FROSCH
290 CALL CHAR(116,"00606600036C363030318
9B6CFFFF00004E80300C6C0066E066625E66FFFF
")!WASSERSPRITZER
300 CALL CHAR(128,"0010111A5ADFFFFFFFFFF
FFFFFFFFFFF")!GRASHALME
310 CALL CHAR(42,"")
320 CALL CHAR(48,"")
330 CALL CHAR(49,"80FF8080FF808080")!LEI
TER TEIL LINKS
340 CALL CHAR(50,"01FF0101FF010101")!LEI
TER TEIL RECHTS
350 CALL CHAR(43,"8080808080808080010101
0101010101 01")!LEITER UEBER WASSER
360 CALL CHAR(140,"00000000031F7F7F7F7F7
F0303000000000000000C0E0F0F8FCFCFCFCF8F8"
)!BLUTFLECK
370 CALL SCREEN(9)
380 CALL COLOR(1,6,12,13,13,8)!SEEFARBE
390 CALL COLOR(2,5,12,3,4,5,4,8,8,12,4,5
)
400 CALL HCHAR(1,1,56,352):: CALL HCHAR(
12,1,42,224):: CALL HCHAR(19,1,48,192)
410 !LEITER
420 CALL VCHAR(18,8,43):: CALL VCHAR(18,
9,44):: CALL VCHAR(19,8,49,2):: CALL VCH
AR(19,9,50,2)
430 !ZEICHNEN SEE
440 CALL HCHAR(13,12,34):: CALL HCHAR(13
,13,33):: CALL HCHAR(13,14,38,19):: CALL
 HCHAR(14,11,35):: CALL HCHAR(14,12,38,2
1)
450 CALL HCHAR(15,11,38,22):: CALL HCHAR
(16,11,38,22):: CALL HCHAR(17,11,36):: C
ALL HCHAR(17,12,38,21):: CALL HCHAR(18,1
2,37)
460 CALL HCHAR(18,13,38,20)
470 CALL HCHAR(11,1,128,32):: CALL HCHAR
(12,1,129,64)
480 CALL HCHAR(23,28,52):: CALL HCHAR(24
,28,53):: CALL HCHAR(23,29,54):: CALL HC
HAR(24,29,55)
490 CALL HCHAR(23,15,120):: CALL HCHAR(2
```

Scan: S. Höltgen


```
4,15,121):: CALL HCHAR(23,16,122):: CALL
 HCHAR(24,16,123)
500 CALL HCHAR(23,5,124):: CALL HCHAR(24
,5,125):: CALL HCHAR(23,6,126):: CALL HC
HAR(24,6,127)
510 CALL SPRITE(#5,96,11,174,20,#6,96,15
,174,140)
520 CALL SPRITE(#9,92,9,158,150,#10,92,1
6,158,250)
530 CALL SPRITE(#11,88,8,142,70,#12,88,1
1,142,180)
540 CALL SPRITE(#1,68,16,5,10,0,-4,#2,72
,16,7,180,0,-4)
545 CALL SPRITE(#16,112,4,110,60)
550 CALL MOTION(#5,0,-10-(2*Q),#6,0,-10-
(2*Q))
560 CALL MOTION(#9,0,10+Q,#10,0,10+Q)
570 CALL MOTION(#11,0,-15-Q,#12,0,-15-Q)
:: GOTO 607
600 IF GEFRESSEN/10=INT(GEFRESSEN/10)THE
N Q=INT(GEFRESSEN/10)ELSE 607
602 GOTO 550
606 REM HOEHE UND GESCHWINDIGKEIT DES MO
SKITO
607 CALL SOUND(300,-8,0,30000,30,30000,3
0,390,0):: MOSKITO=MOSKITO+1 :: CALL SPR
ITE(#17,80,2,INT(RND*62)+10,240,0,-(INT(
RND*20)+4))
610 REM ABSPRUNG FROSCH
620 CALL KEY(2,K,S):: IF K=18 THEN 630 E
LSE 620
630 CALL MOTION(#16,-10,10):: CALL PATTE
RN(#16,104):: CALL SOUND(50,400,2,410,2)
:: CALL SOUND(50,300,2,310,2)
640 CALL COINC(#16,#17,10,B):: IF B=-1 T
HEN 670 ELSE CALL POSITION(#16,X,Y)
650 IF X<15 THEN 710 ELSE 640
660 REM MOSKITO ERREICHT
670 CALL MOTION(#16,0,0):: CALL DELSPRIT
E(#17):: FOR I=1 TO 7 :: CALL SOUND(100,
300+I*100,2):: NEXT I :: GEFRESSEN=GEFRE
SSEN+1
680 REM NEUE POSITION FUER FROSCH
690 CALL MOTION(#16,0,0):: CALL DELSPRIT
E(#16):: CALL SPRITE(#16,112,3,INT(RND*3
0)+100,INT(RND*45)+24):: CALL SOUND(200,
-1,0):: GOTO 600
700 REM ABSTURZ FROSCH
710 CALL MOTION(#16,0,0):: CALL PATTERN(
#16,108):: CALL COLOR(#16,10):: CALL MOT
ION(#16,8,2):: CALL DELSPRITE(#17)
720 CALL SOUND(100,110,0):: CALL POSITIO
N(#16,X,Y):: IF X>115 THEN CALL MOTION(#
16,0,0)ELSE 720
730 REM WASSERSPRITZER
740 CALL PATTERN(#16,116):: CALL SOUND(5
00,-7,0):: FOR L=1 TO 130 :: NEXT L :: C
ALL DELSPRITE(#16)
750 REM FROSCH IM WASSER
760 CALL SPRITE(#16,100,16,182,190)
770 CALL SOUND(100,1000,0):: CALL KEY(2,
K,S):: IF K<>18 THEN 770
780 CALL POSITION(#16,A,G):: CALL JOYST(
2,X,Y):: CALL MOTION(#16,-Y*2,X*2):: IF
A>185 OR A<144 THEN 830
790 CALL COINC(ALL,B):: IF B=-1 THEN 830
 ELSE CALL DISTANCE(#16,153,61,H)
800 REM LEITER ERREICHT
810 IF H<90 THEN 900 ELSE 780
820 REM FROSCH VON TIER GEFRESSEN
830 CALL COLOR(#16,9)
840 CALL PATTERN(#16,140):: CALL MOTION(
#16,0,0):: CALL SOUND(100,-5,0):: FOR D=
1 TO 400 :: NEXT D
850 CALL DELSPRITE(#16):: CALL COLOR(#16
,2):: FROSCH=FROSCH-1 :: IF FROSCH=0 THE
N 860 ELSE GOTO 690
860 FOR X=780 TO 300 STEP -5 :: CALL SOU
ND(-80,X,0,X+10,2,X+20,4):: NEXT X
870 CALL DELSPRITE(ALL):: CALL CLEAR ::
CALL CHARSET
880 DISPLAY AT(9,1):"VON";MOSKITO;"MOSKI
TOS HABEN SIE":GEFRESSEN;"GEFRESSEN!!!"
885 DISPLAY AT(16,1):"DAS SIND";INT(GEFR
ESSEN/MOSKITO*100);"% !!"
890 DISPLAY AT(23,1):"NOCH EIN SPIEL (J/
N)?" :: ACCEPT AT(23,22):A$ :: IF A$="J"
 OR A$="j" THEN 10 ELSE STOP
900 CALL MOTION(#16,0,0):: FOR D=1 TO 10
 :: CALL SOUND(50,300,0,310,0):: CALL SO
UND(50,450,0,460,0):: NEXT D :: GOTO 690
```

Alien

Mit Luftbomben versucht ein Helikopter, ein Hochhaus vor einer heranfliegenden Untertasse zu schützen. Eindringlicher Sound und große Spielfiguren beleben die Szene auf dem C 64 mit Joystick

Der Joystick wird an PORT 2 angeschlossen. Nach dem Laden des Programmes wird es mit RUN gestartet. Es folgt darauf eine Abfrage, ob ein Schwarzweiß- oder ein Farbfernsehgerät am Rechner angeschlossen ist. Hat man den entsprechenden Buchstaben eingegeben, so erscheint das Titelbild am Bildschirm. Durch Drücken einer Taste wird das Spiel gestartet.

Geschwindigkeit bestimmt Punktzahl

Zuerst erscheint die Rundenanzeige, begleitet von einer Melodie. Anschließend bekommt man das Spielfeld zu sehen. Ziel des Spieles ist es nun, heranfliegende Untertassen mittels Luftbomben zu zerstören. In der ersten Runde besitzt man zehn Bomben, in der zweiten Runde zwanzig

```
350 :
360 :
361 :
380 REM LANDSCHAFT
390 :
391 REM DEFINITION VON BEREICH 1
400 SDEF 1
410 SD 000000000000000000000000
420 SD 000000000000000000000000
430 SD 000000000000000000000000
440 SD 000000000000000000000000
450 SD 000000000000000000000000
460 SD 000000000000000000000000
470 SD 000000000000000000000000
480 SD 000000000000000000000000
490 SD 000000000000000000000000
500 SD 000000000000000000000000
510 SD 000000000000000000000000
520 SD 000000000000000000000000
530 SD 000000000001011000001100
540 SD 110011000111001110011101
550 SD 110111010011001100000111
560 SD 011110110011011001101110
570 SD 000110101111011101100111
580 SD 110111100111111100111111
590 SD 011111000001100000011110
600 SD 000110000011110000001110
610 SD 111111111111111111111111
620 :
630 REM INITIALISIERUNG DER SPRITES 0-4
631 REM WIESE
640 SPRITE 0,1,1,1,2,1,3,1,4,1,5,1
650 YCOR 0,170,1,170,2,170,3,170,4,170,5
,170
660 XCOR 0,024,1,075,2,127,3,180,4,236,5
,290
670 COLOR 0,5,1,5,2,5,3,5,4,5,5,5:REM GR
UEN
680 XEXP 0,1,2,3,4,5
690 YEXP 0,1,2,3,4,5
700 :
710 PRINT"";
720 FORI=1TO80:PRINT" ";:NEXT
730 :
740 :
750 REM SONNE
760 SDEF 2
770 SD 000000001100011000000000
780 SD 000000001100011000000000
790 SD 000110001100011000110000
800 SD 000111000110110001110000
810 SD 000011100110110011100000
820 SD 000001110111110111000000
830 SD 000000111111111110000000
840 SD 011100011111111100011100
850 SD 000011111111111111100000
860 SD 000000111111111110000000
870 SD 000000111111111110000000
880 SD 000011111111111111100000
890 SD 011100011111111100011100
900 SD 000000111111111110000000
910 SD 000001110111110111000000
920 SD 000011100110110011100000
930 SD 000111000110110001110000
940 SD 000110001100011000110000
950 SD 000000001100011000000000
960 SD 000000001100011000000000
970 SD 000000000000000000000000
980 :
990 REM INITIALISIERUNG VON SPRITE 6
1000 REM SONNE
1010 SPRITE 6,2
1020 YCOR 6,50
1030 XCOR 6,240
1040 COLOR 6,7:REM GELB
1050 XEXP 6
1060 YEXP 6
1070 RETURN

READY.
```

Demonstrationsprogramm zur Benutzung von Sprite Master

Balken-Diagramm

In vertikaler oder horizontaler Balkenform können auf dem TI-99/4A in der Grundversion Zahlenwerte gegenübergestellt werden

Die vertikale Version ermöglicht die Darstellung von zwölf Zahlen in den Werten zwischen 1 bis 20, die horizontale Version dagegen die Darstellung von zehn Zahlen in den Werten zwischen 1 bis 25.

Das Programm rundet

Zahlen, die größer oder kleiner als der erlaubte Zahlenumfang sind, werden vom Computer auf- oder abgerundet, ebenso die Zwischenwerte.

Gibt man bei einer Vergleichszahl-Abfrage die Ziffer 0 ein, so wird in diesem Feld kein Balken ausgedruckt. Hat man also wenige Zahlen zu vergleichen, gibt man einfach bei den restlichen Zahlenabfragen 0 ein.

Ein Menü hilft bei der Bedienung

Nach Auswahl der gewünschten Programmversion (horizontal oder vertikal) und nach Eingabe der Vergleichszahlen wird auf dem Bildschirm das Balkendiagramm ausgedruckt. Man sieht den Balken allmählich wachsen.

```
100 REM ******************
110 REM *                *
120 REM * balkendiagramm *
130 REM *                *
140 REM *                *
150 REM *  programm von  *
160 REM *                *
170 REM *  hans-juergen  *
180 REM *                *
190 REM *    sawinski    *
200 REM *                *
210 REM *   dornstr. 3   *
220 REM *                *
230 REM *  4690 herne 1  *
240 REM *                *
250 REM ******************
260 REM
270 REM def. sonderzeichen

280 CALL CHAR(128,"FFFFFFFF
FFFFFFF")
290 CALL CHAR(129,"FFFF01010
```

Scan: S. Höltgen

Nach dem Ausdruck auf dem Bildschirm und nach einer kurzen Warteschleife fordert der Computer zum Weitergehen auf. Nach dem Drücken der Space-Taste werden die möglichen nächsten Schritte des Anwenders angezeigt. Hierbei hilft ein komfortables Menü. *Hans-Jürgen Sawinski*

```
1010101")
300 CALL CHAR(130,"FF8080808
0808080")
310 CALL CHAR(131,"010101010
1010101")
320 CALL CHAR(132,"FF8181818
1818181")
330 CALL CHAR(133,"FFFF01010
10101FF")
340 CALL CLEAR
350 PRINT "  **** BALKENDIAG
RAMM ****  "
360 PRINT
370 PRINT "  *** FUER DEN TI
99/4A ***"
380 PRINT ::::::::::
390 FOR VERZ=1 TO 2000
400 NEXT VERZ
410 REM
420 REM  menue
430 REM
440 CALL CLEAR
450 PRINT TAB(6);"** AUSWAHL
MENUE **"
460 PRINT TAB(6);"==========
========"
470 PRINT ::
480 PRINT TAB(6);"1) DIAGRAM
M TYP A"
490 PRINT TAB(6);"   MIT ANM
ERKUNG"
500 PRINT
510 PRINT TAB(6);"2) DIAGRAM
M TYP A"
520 PRINT TAB(6);"   OHNE AN
MERKUNG"
530 PRINT
540 PRINT TAB(6);"3) DIAGRAM
M TYP B"
550 PRINT TAB(6);"   MIT ANM
ERKUNG"
560 PRINT
570 PRINT TAB(6);"4) DIAGRAM
M TYP B"
580 PRINT TAB(6);"   OHNE AN
MERKUNG"
590 PRINT ::
600 CALL KEY(0,KEY,STATUS)
610 IF KEY<>49 THEN 630
620 GOTO 690
630 IF KEY<>50 THEN 650
640 GOTO 960
650 IF KEY<>51 THEN 670
660 GOTO 3100
670 IF KEY<>52 THEN 600
680 GOTO 3370
690 CALL CLEAR
700 REM
710 REM  diagramm typ a
720 REM  anleitung
730 REM
740 PRINT TAB(8);"DIAGRAMM T
YP A"
750 PRINT TAB(8);"==========
====":: 
760 PRINT "MIT DIESEM PROGRA
MM KOENNEN"
770 PRINT "MITEINANDER ZU VE
RGLEICHEN-"
780 PRINT "DE ZAHLEN GRAPHIS
CH DARGE- "
790 PRINT "STELLT WERDEN."
800 PRINT
810 PRINT "ES KOENNEN BIS ZU
 10 VER-"
820 PRINT "SCHIEDENE ZAHLEN
MITEINAN-"
830 PRINT "DER VERGLICHEN WE
RDEN."
840 PRINT "DER COMPUTER STEL
LT GANZE"
850 PRINT "ZAHLEN ZWISCHEN 1
 BIS 25 "
860 PRINT "GRAPHISCH DAR (HO
RIZONTAL).":: 
870 PRINT "ZAHLEN, DIE GROES
SER ALS 25,"
880 PRINT "KLEINER ALS 1 ODE
R BRUCH-"
890 PRINT "WERTE SIND, RUNDE
T DER COM-"
900 PRINT "PUTER AUTOMATISCH
 AUF ODER"
910 PRINT "AB.":: 
920 PRINT "ZUM PROGRAMMSTART
 BITTE NUN"
930 PRINT "DIE SPACE-TASTE D
RUECKEN!"
940 CALL KEY(0,KEY,STATUS)
950 IF KEY<>32 THEN 940
960 CALL CLEAR
970 REM
980 REM  vz-abfrage
990 REM
1000 PRINT "   *** DIAGRAMM
TYP A ***"
1010 PRINT "   =============
=========="
1020 PRINT ::
1030 INPUT "1. VERGLEICHSZAH
L: ":D
1040 IF D=0 THEN 1090
1050 IF D>25 THEN 1060 ELSE
1070
1060 GOSUB 2700
1070 IF D<1 THEN 1080 ELSE 1
090
1080 GOSUB 2720
1090 D=D+5
1100 PRINT
1110 INPUT "2. VERGLEICHSZAH
L: ":E
1120 IF E=0 THEN 1170
1130 IF E>25 THEN 1140 ELSE
1150
1140 GOSUB 2740
1150 IF E<1 THEN 1160 ELSE 1
170
1160 GOSUB 2760
1170 E=E+5
1180 PRINT
1190 INPUT "3. VERGLEICHSZAH
L: ":F
1200 IF F=0 THEN 1250
1210 IF F>25 THEN 1220 ELSE
1230
1220 GOSUB 2780
1230 IF F<1 THEN 1240 ELSE 1
250
1240 GOSUB 2800
1250 F=F+5
1260 PRINT
1270 INPUT "4. VERGLEICHSZAH
L: ":G
1280 IF G=0 THEN 1330
1290 IF G>25 THEN 1300 ELSE
1310
1300 GOSUB 2820
1310 IF G<1 THEN 1320 ELSE 1
330
1320 GOSUB 2840
1330 G=G+5
1340 PRINT
1350 INPUT "5. VERGLEICHSZAH
L: ":H
1360 IF H=0 THEN 1410
1370 IF H>25 THEN 1380 ELSE
1390
1380 GOSUB 2860
1390 IF H<1 THEN 1400 ELSE 1
410
1400 GOSUB 2880
1410 H=H+5
1420 PRINT
1430 INPUT "6. VERGLEICHSZAH
L: ":I
1440 IF I=0 THEN 1490
1450 IF I>25 THEN 1460 ELSE
1470
1460 GOSUB 2900
1470 IF I<1 THEN 1480 ELSE 1
490
1480 GOSUB 2920
1490 I=I+5
1500 PRINT
1510 INPUT "7. VERGLEICHSZAH
L: ":J
1520 IF J=0 THEN 1570
1530 IF J>25 THEN 1540 ELSE
1550
1540 GOSUB 2940
1550 IF J<1 THEN 1560 ELSE 1
570
1560 GOSUB 2960
1570 J=J+5
1580 PRINT
1590 INPUT "8. VERGLEICHSZAH
L: ":K
1600 IF K=0 THEN 1650
1610 IF K>25 THEN 1620 ELSE
1630
1620 GOSUB 2980
1630 IF K<1 THEN 1640 ELSE 1
650
1640 GOSUB 3000
1650 K=K+5
1660 PRINT
1670 INPUT "9. VERGLEICHSZAH
L: ":L
1680 IF L=0 THEN 1730
1690 IF L>25 THEN 1700 ELSE
1710
1700 GOSUB 3020
1710 IF L<1 THEN 1720 ELSE 1
730
1720 GOSUB 3040
1730 L=L+5
1740 PRINT
1750 INPUT "10. VERGLEICHSZA
HL: ":M
1760 IF M=0 THEN 1810
1770 IF M>25 THEN 1780 ELSE
1790
1780 GOSUB 3060
1790 IF M<1 THEN 1800 ELSE 1
810
1800 GOSUB 3080
1810 M=M+5
1820 REM
1830 REM  vert. zahlenreihe

1840 REM
1850 CALL CLEAR
1860 CALL VCHAR(2,4,49)
1870 CALL VCHAR(4,4,50)
1880 CALL VCHAR(6,4,51)
1890 CALL VCHAR(8,4,52)
1900 CALL VCHAR(10,4,53)
1910 CALL VCHAR(12,4,54)
1920 CALL VCHAR(14,4,55)
1930 CALL VCHAR(16,4,56)
1940 CALL VCHAR(18,4,57)
1950 CALL VCHAR(20,3,49)
1960 CALL VCHAR(20,4,48)
1970 CALL VCHAR(2,5,133)
1980 REM
1990 REM  y-achse
2000 REM
2010 FOR Z=2 TO 21
2020 CALL VCHAR(Z,5,129)
2030 NEXT Z
2040 CALL VCHAR(22,5,131)
2050 REM
2060 REM  x-achse
2070 REM
2080 CALL HCHAR(22,6,130,2)
2090 FOR Z=7 TO 29
2100 CALL HCHAR(22,Z,130)
2110 NEXT Z
2120 CALL HCHAR(22,30,132)
2130 REM
2140 REM  horiz.zahlenreihe

2150 REM
2160 CALL HCHAR(23,5,48)
2170 CALL HCHAR(23,6,48)
2180 CALL HCHAR(23,10,48)
2190 CALL HCHAR(23,11,53)
2200 CALL HCHAR(23,15,49)
2210 CALL HCHAR(23,16,48)
2220 CALL HCHAR(23,20,49)
2230 CALL HCHAR(23,21,53)
2240 CALL HCHAR(23,25,50)
2250 CALL HCHAR(23,26,48)
2260 CALL HCHAR(23,30,50)
2270 CALL HCHAR(23,31,53)
2280 REM
2290 REM  balkendarstellung
```

Scan: S. Höltgen


```
2300 REM
2310 FOR Z=6 TO D
2320 CALL HCHAR(2,Z,128)
2330 NEXT Z
2340 FOR Z=6 TO E
2350 CALL HCHAR(4,Z,128)
2360 NEXT Z
2370 FOR Z=6 TO F
2380 CALL HCHAR(6,Z,128)
2390 NEXT Z
2400 FOR Z=6 TO G
2410 CALL HCHAR(8,Z,128)
2420 NEXT Z
2430 FOR Z=6 TO H
2440 CALL HCHAR(10,Z,128)
2450 NEXT Z
2460 FOR Z=6 TO I
2470 CALL HCHAR(12,Z,128)
2480 NEXT Z
2490 FOR Z=6 TO J
2500 CALL HCHAR(14,Z,128)
2510 NEXT Z
2520 FOR Z=6 TO K
2530 CALL HCHAR(16,Z,128)
2540 NEXT Z
2550 FOR Z=6 TO L
2560 CALL HCHAR(18,Z,128)
2570 NEXT Z
2580 FOR Z=6 TO M
2590 CALL HCHAR(20,Z,128)
2600 NEXT Z
2610 FOR Z=1 TO 3000
2620 NEXT Z
2630 PRINT "WEITER MIT SPACE
~TASTE ! ! !"
2640 CALL KEY(0,KEY,STATUS)
2650 IF KEY<>32 THEN 2640
2660 GOTO 5930
2670 REM
2680 REM  werte runden
          <1 oder >25
2690 REM
2700 D=25
2710 RETURN
2720 D=1
2730 RETURN
2740 E=25
2750 RETURN
2760 E=1
2770 RETURN
2780 F=25
2790 RETURN
2800 F=1
2810 RETURN
2820 G=25
2830 RETURN
2840 G=1
2850 RETURN
2860 H=25
2870 RETURN
2880 H=1
2890 RETURN
2900 I=25
2910 RETURN
2920 I=1
2930 RETURN
2940 J=25
2950 RETURN
2960 J=1
2970 RETURN
2980 K=25
2990 RETURN
3000 K=1
3010 RETURN
3020 L=25
3030 RETURN
3040 L=1
3050 RETURN
3060 M=25
3070 RETURN
3080 M=1
3090 RETURN
3100 CALL CLEAR
3110 REM
3120 REM   diagramm typ b
3130 REM   anleitung
3140 REM
3150 PRINT TAB(8);"DIAGRAMM
TYP B"
3160 PRINT TAB(8);"=========
====="::
3170 PRINT "MIT DIESEM PROGR
AMM KOENNEN"
3180 PRINT "MITEINANDER ZU V
ERGLEICHEN-"
3190 PRINT "DE ZAHLEN GRAPHI
SCH DARGE-"
3200 PRINT "STELLT WERDEN."
3210 PRINT
3220 PRINT "ES KOENNEN BIS Z
U 12 VER-"
3230 PRINT "SCHIEDENE ZAHLEN
 MITEINAN-"
3240 PRINT "DER VERGLICHEN W
ERDEN."
3250 PRINT "DER COMPUTER STE
LLT GANZE"
3260 PRINT "ZAHLEN ZWISCHEN
1 BIS 20"
3270 PRINT "GRAPHISCH DAR (V
ERTIKAL)."::
3280 PRINT "ZAHLEN, DIE GROE
SSER ALS 20,"
3290 PRINT "KLEINER ALS 1 OD
ER BRUCH-"
3300 PRINT "WERTE SIND, RUND
ET DER COM-"
3310 PRINT "PUTER AUTOMATISC
H AUF ODER"
3320 PRINT "AB."::
3330 PRINT "ZUM PROGRAMMSTAR
T BITTE NUN"
3340 PRINT "DIE SPACE~TASTE
DRUECKEN!"
3350 CALL KEY(0,KEY,STATUS)
3360 IF KEY<>32 THEN 3350
3370 CALL CLEAR
3380 REM
3390 REM  vz-abfrage
3400 REM
3410 PRINT "    *** DIAGRAMM
TYP B ***"
3420 PRINT "    =============
========"
3430 PRINT ::
3440 INPUT "1. VERGLEICHSZAH
L: ":A
3450 IF A=0 THEN 3500
3460 IF A>20 THEN 3470 ELSE
3480
3470 GOSUB 5420
3480 IF A<1 THEN 3490 ELSE 3
500
3490 GOSUB 5440
3500 A=22-A
3510 PRINT
3520 INPUT "2. VERGLEICHSZAH
L: ":B
3530 IF B=0 THEN 3580
3540 IF B>20 THEN 3550 ELSE
3560
3550 GOSUB 5460
3560 IF B<1 THEN 3570 ELSE 3
580
3570 GOSUB 5480
3580 B=22-B
3590 PRINT
3600 INPUT "3. VERGLEICHSZAH
L: ":C
3610 IF C=0 THEN 3660
3620 IF C>20 THEN 3630 ELSE
3640
3630 GOSUB 5500
3640 IF C<1 THEN 3650 ELSE 3
660
3650 GOSUB 5520
3660 C=22-C
3670 PRINT
3680 INPUT "4. VERGLEICHSZAH
L: ":D
3690 IF D=0 THEN 3740
3700 IF D>20 THEN 3710 ELSE
3720
3710 GOSUB 5540
3720 IF D<1 THEN 3730 ELSE 3
740
3730 GOSUB 5560
3740 D=22-D
3750 PRINT
3760 INPUT "5. VERGLEICHSZAH
L: ":E
3770 IF E=0 THEN 3820
3780 IF E>20 THEN 3790 ELSE
3800
3790 GOSUB 5580
3800 IF E<1 THEN 3810 ELSE 3
820
3810 GOSUB 5600
3820 E=22-E
3830 PRINT
3840 INPUT "6. VERGLEICHSZAH
L: ":F
3850 IF F=0 THEN 3900
3860 IF F>20 THEN 3870 ELSE
3880
3870 GOSUB 5620
3880 IF F<1 THEN 3890 ELSE 3
900
3890 GOSUB 5640
3900 F=22-F
3910 PRINT
3920 INPUT "7. VERGLEICHSZAH
L: ":G
3930 IF G=0 THEN 3980
3940 IF G>20 THEN 3950 ELSE
3960
3950 GOSUB 5660
3960 IF G<1 THEN 3970 ELSE 3
980
3970 GOSUB 5680
3980 G=22-G
3990 PRINT
4000 INPUT "8. VERGLEICHSZAH
L: ":H
4010 IF H=0 THEN 4060
4020 IF H>20 THEN 4030 ELSE
4040
4030 GOSUB 5700
4040 IF H<1 THEN 4050 ELSE 4
060
4050 GOSUB 5720
4060 H=22-H
4070 PRINT
4080 INPUT "9. VERGLEICHSZAH
L: ":I
4090 IF I=0 THEN 4140
4100 IF I>20 THEN 4110 ELSE
4120
4110 GOSUB 5740
4120 IF I<1 THEN 4130 ELSE 4
140
4130 GOSUB 5760
4140 I=22-I
4150 PRINT
4160 INPUT "10.VERGLEICHSZAH
L: ":J
4170 IF J=0 THEN 4220
4180 IF J>20 THEN 4190 ELSE
4200
4190 GOSUB 5780
4200 IF J<1 THEN 4210 ELSE 4
220
4210 GOSUB 5800
4220 J=22-J
4230 PRINT
4240 INPUT "11.VERGLEICHSZAH
L: ":K
4250 IF K=0 THEN 4300
4260 IF K>20 THEN 4270 ELSE
4280
4270 GOSUB 5820
4280 IF K<1 THEN 4290 ELSE 4
300
4290 GOSUB 5840
4300 K=22-K
4310 PRINT
4320 INPUT "12.VERGLEICHSZAH
L: ":L
4330 IF L=0 THEN 4380
4340 IF L>20 THEN 4350 ELSE
4360
4350 GOSUB 5860
4360 IF L<1 THEN 4370 ELSE 4
380
4370 GOSUB 5880
4380 L=22-L
4390 REM
4400 REM  vert. zahlenreihe

4410 REM
4420 CALL CLEAR
4430 CALL VCHAR(2,3,50)
4440 CALL VCHAR(2,4,48)
4450 CALL VCHAR(4,3,49)
4460 CALL VCHAR(4,4,56)
4470 CALL VCHAR(6,3,49)
4480 CALL VCHAR(6,4,54)
4490 CALL VCHAR(8,3,49)
4500 CALL VCHAR(8,4,52)
4510 CALL VCHAR(10,3,49)
4520 CALL VCHAR(10,4,50)
4530 CALL VCHAR(12,3,49)
4540 CALL VCHAR(12,4,48)
4550 CALL VCHAR(14,4,56)
```

Scan: S. Höltgen


```
4560 CALL VCHAR(16,4,54)
4570 CALL VCHAR(18,4,52)
4580 CALL VCHAR(20,4,50)
4590 CALL VCHAR(2,5,133)
4600 REM
4610 REM  y-achse
4620 REM
4630 FOR Z=2 TO 21
4640 CALL VCHAR(Z,5,129)
4650 NEXT Z
4660 CALL VCHAR(22,5,131)
4670 REM
4680 REM  x-achse
4690 REM
4700 CALL HCHAR(22,6,130,2)
4710 FOR Z=7 TO 29
4720 CALL HCHAR(22,Z,130)
4730 NEXT Z
4740 CALL HCHAR(22,30,132)
4750 REM
4760 REM  horiz.zahlenreihe

4770 REM
4780 CALL HCHAR(23,5,48)
4790 CALL HCHAR(23,7,49)
4800 CALL HCHAR(23,9,50)
4810 CALL HCHAR(23,11,51)
4820 CALL HCHAR(23,13,52)
4830 CALL HCHAR(23,15,53)
4840 CALL HCHAR(23,17,54)
4850 CALL HCHAR(23,19,55)
4860 CALL HCHAR(23,21,56)
4870 CALL HCHAR(23,23,57)
4880 CALL HCHAR(23,25,49)
4890 CALL HCHAR(23,26,48)
4900 CALL HCHAR(23,27,49)
4910 CALL HCHAR(23,28,49)
4920 CALL HCHAR(23,29,49)
4930 CALL HCHAR(23,30,50)
4940 REM
4950 REM  balkendarstellung

4960 REM
4970 FOR Z=21 TO A STEP -1
4980 CALL VCHAR(Z,7,128)
4990 NEXT Z
5000 FOR Z=21 TO B STEP -1
5010 CALL VCHAR(Z,9,128)
5020 NEXT Z
5030 FOR Z=21 TO C STEP -1
5040 CALL VCHAR(Z,11,128)
5050 NEXT Z
5060 FOR Z=21 TO D STEP -1
5070 CALL VCHAR(Z,13,128)
5080 NEXT Z
5090 FOR Z=21 TO E STEP -1
5100 CALL VCHAR(Z,15,128)
5110 NEXT Z
5120 FOR Z=21 TO F STEP -1
5130 CALL VCHAR(Z,17,128)
5140 NEXT Z
5150 FOR Z=21 TO G STEP -1
5160 CALL VCHAR(Z,19,128)
5170 NEXT Z
5180 FOR Z=21 TO H STEP -1
5190 CALL VCHAR(Z,21,128)
5200 NEXT Z
5210 FOR Z=21 TO I STEP -1
5220 CALL VCHAR(Z,23,128)
5230 NEXT Z
5240 FOR Z=21 TO J STEP -1
5250 CALL VCHAR(Z,25,128)
5260 NEXT Z
5270 FOR Z=21 TO K STEP -1
5280 CALL VCHAR(Z,27,128)
5290 NEXT Z
5300 FOR Z=21 TO L STEP -1
5310 CALL VCHAR(Z,29,128)
5320 NEXT Z
5330 FOR Z=1 TO 3000
5340 NEXT Z
5350 PRINT "WEITER MIT SPACE
~TASTE ! ! !"
5360 CALL KEY(0,KEY,STATUS)
5370 IF KEY<>32 THEN 5360
5380 GOTO 5930
5390 REM
5400 REM  runden werte
          <1 oder >20

5410 REM
5420 A=20
5430 RETURN
5440 A=1
5450 RETURN
5460 B=20
5470 RETURN
5480 B=1
5490 RETURN
5500 C=20
5510 RETURN
5520 C=1
5530 RETURN
5540 D=20
5550 RETURN
5560 D=1
5570 RETURN
5580 E=20
5590 RETURN
5600 E=1
5610 RETURN
5620 F=20
5630 RETURN
5640 F=1
5650 RETURN
5660 G=20
5670 RETURN
5680 G=1
5690 RETURN
5700 H=20
5710 RETURN
5720 H=1
5730 RETURN
5740 I=20
5750 RETURN
5760 I=1
5770 RETURN
5780 J=20
5790 RETURN
5800 J=1
5810 RETURN
5820 K=20
5830 RETURN
5840 K=1
5850 RETURN
5860 L=20
5870 RETURN
5880 L=1
5890 RETURN
5900 REM
5910 REM  abschluss
5920 REM
5930 CALL CLEAR
5940 PRINT "NAECHSTER SCHRIT
T VON IHNEN?"
5950 PRINT "================
============="
5960 PRINT ::
5970 PRINT "1) ZURUECK ZUM P
ROGRAMMKOPF"
5980 PRINT
5990 PRINT "2) VERGLEICH DIA
GRAMM TYP A"
6000 PRINT "   MIT ANMERKUNG
"
6010 PRINT
6020 PRINT "3) VERGLEICH DIA
GRAMM TYP A"
6030 PRINT "   OHNE ANMERKUN
G"
6040 PRINT
6050 PRINT "4) VERGLEICH DIA
GRAMM TYP B"
6060 PRINT "   MIT ANMERKUNG
"
6070 PRINT
6080 PRINT "5) VERGLEICH DIA
GRAMM TYP B"
6090 PRINT "   OHNE ANMERKUN
G"
6100 PRINT
6110 PRINT "6) PROGRAMM~MODU
S VERLASSEN"
6120 PRINT
6130 CALL KEY(0,KEY,STATUS)
6140 IF KEY<>49 THEN 6160
6150 GOTO 280
6160 IF KEY<>50 THEN 6180
6170 GOTO 690
6180 IF KEY<>51 THEN 6200
6190 GOTO 960
6200 IF KEY<>52 THEN 6220
6210 GOTO 3100
6220 IF KEY<>53 THEN 6240
6230 GOTO 3370
6240 IF KEY<>54 THEN 6130
6250 CALL CLEAR
6260 END
```

Snake

Eine Schlange frißt Zahlen auf dem Bildschirm und wird länger und länger (Sharp MZ-700)

Der Fraß der Schlange besteht aus Zahlen zwischen Eins und Neun. Die gefressene Zahl gibt an, um wieviel Glieder die Schlange wächst.

Ziel des Spielers ist es, die Schlange so geschickt zu dirigieren, daß sie einen möglichst langen Schwanz hinter sich herzieht. Ein Spiel ist zu Ende, wenn die Schlange gegen die Randbegrenzung, zwei Kreuze in der Mitte oder eines seiner Schwanzteile läuft.

Es sind drei Schwierigkeitsgrade wählbar, die die Intervalle angeben, in denen sich die Kreuze verändern. Das wirkt sich jedoch erst aus, wenn der Schwanz schon sehr lang ist oder wenn außerdem die Zufallszahl in der Nähe der Kreuze positioniert wurde.

Bei den PRINT-Anweisungen ist darauf zu achten, daß bei der Eingabe des Listings kein Semikolon vergessen wird.

Riko Wichmann

Buck Rogers

Kampf im Weltraum

Der für Computer-Spieler fast schon „gute, alte Weltraum" mußte wieder einmal herhalten, um für Spannung auf dem Bildschirm zu sorgen. Es hat ja in letzter Zeit eine Vielzahl von Weltraumspielen gegeben, die teilweise so stupide angelegt waren, daß man sich in Opas Pantoffelkino wähnte. Bei diesem Spiel nehmen Sie die Rolle von Buck Rogers, dem Kommandanten eines riesigen Raumschiffes, ein, das auf Erkundungsflug über dem Planeten Zoom angelangt ist. Der Auftrag war geheim. Sie konnten sich deshalb auch nicht über alle Eventualitäten informieren. Bekannt ist Ihnen nur, daß es in dieser Gegend von feindlichen Schiffen nur so wimmelt. Deshalb hat Ihnen auch Ihre Base drei Raumschiffe bereitgestellt. Sie müssen nun möglichst unauffällig Ihr Raumschiff durch die elektronischen Vorposten steuern. Auf Ihrem Weg werden Ihne viele feindliche Raumschiffe begegnen, die es zu vernichten gilt. Wenn Sie einen neuen „Level" erreicht haben, wechselt der Planet Zoom seine Farbe und die elektronischen Vorposten rükken enger zusammen.

Sie müssen dann durch mehrere dieser Vorposten fliegen und auch mehr feindliche Flugkörper abschießen, um Ihr Mutterschiff wieder zu erreichen. Außerdem erhöht sich die Fluggeschwindigkeit. Sobald Sie mit einem dieser Vorposten kollidieren, verlieren Sie Ihr Schiff. Merke: Sie haben nur drei Raumschiffe zur Verfügung und müssen deshalb sorgfältig Ihre Flugmanöver ausführen.

Schwierigkeitsgrad: Er steigt von „Level" zu „Level".

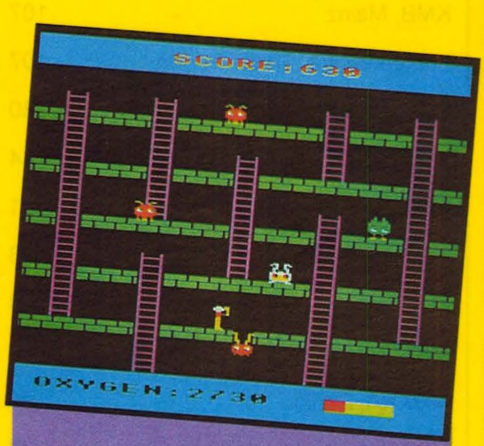
Spieler: 1

Unser Testurteil: Dieses Spiel wird erst auf dem zweiten Blick interessant. Am Anfang sitzt man etwas verwirrt vor dem Bildschirm und findet sich nur schwer mit dem Geschehen zurecht. Die Anleitung ist etwas dürftig. Außerdem liegt sie in englischer Sprache vor. Wir hätten uns gerade bei der Beschreibung des Spiels mehr Hilfestellung erwünscht. So muß man sich also erst langsam in das Spiel hineinarbeiten. Wenn diese Anfangshemmschwelle überwunden ist, dann bekommt man allerdings Spaß. Und den nicht zu wenig. Wir meinen sogar, daß der Hersteller Buck Rogers etwas unter Wert verkauft.

Hersteller: Sega für Texas Instruments im Vertrieb von Radix Bürotechnik, Bornstraße 4, 2000 Hamburg 13

Geeignet: für TI-99/4A

Preis: 99 Mark



Monsters

Maus im Pelz

Da hat man es nun: Seit sich die Computer-Welt für Labyrinthspiele interessiert hat, weiß die Branche diese Marktlücke zu nutzen. Kaum ein Home-Computer, für den es nicht auch ein Labyrinthspiel gibt. Und was einmal erfolgreich als „PacMan" begonnen wurde zieht immer weitere Kreise. Wenn man dazu noch einen weiteren Erfolg dazunimmt, dann dürfte der Marketing-Erfolg sicher sein. Das Spiel, von dem hier die Rede ist, nennt sich „Monsters" und ist für den Acorn-Model B geschrieben. Dieses Modell wird bekanntlich in England unter dem Namen der berühmten Fernsehgesellschaft BBC verkauft. Eine erfolgreiche Fernsehreihe über Computer hat den Acorn-Model B zu einem beliebten Star gemacht.

Stellen Sie sich also Monsters vor – liebe und böse, die bösen jagen das liebe. Was bei PacMan noch in einem Labyrinth passiert, ist hier in eine Kombination aus Gerüst und Labyrinth verlagert worden. Gezählt wird nach Punkten, wenn die wilde Monsterjagd beginnt. Verfolgt von Monsters entlang von Mauern und auf Leitern sind Sie als Computer-Spieler ständig auf der Flucht. Ihre Chance besteht darin, die Monster in Löcher fallen zu lassen. Nur: Das funktioniert, wenn Sie sofort nach einem „Durchfall" Ihrer Verfolger das Loch zumachen. Wenn nicht, dann beginnt die Jagd gleich wieder und Sie sind sich wieder Ihres Lebens nicht sicher. Je länger Sie den Monstern entgehen, um so mehr Punkte werden Sie sammeln. Also auf zur wilden Monsterjagd.

Schwierigkeitsgrad: Steigerung mit dem Fortgang des Spiels, wobei die Jagd immer gefährlicher wird.

Spieler: 1

Unser Testurteil: Wir haben schon viele derartige Spiele gesehen. Sie sind eben jetzt Mode. Acorn ist es gelungen, einige Spannung in das Programm zu integrieren. Gut gelungen scheint uns auch die Ausnutzung der vorhandenen Speicherkapazität zu sein. Die Szenen erscheinen klar und deutlich auf dem Bildschirm.

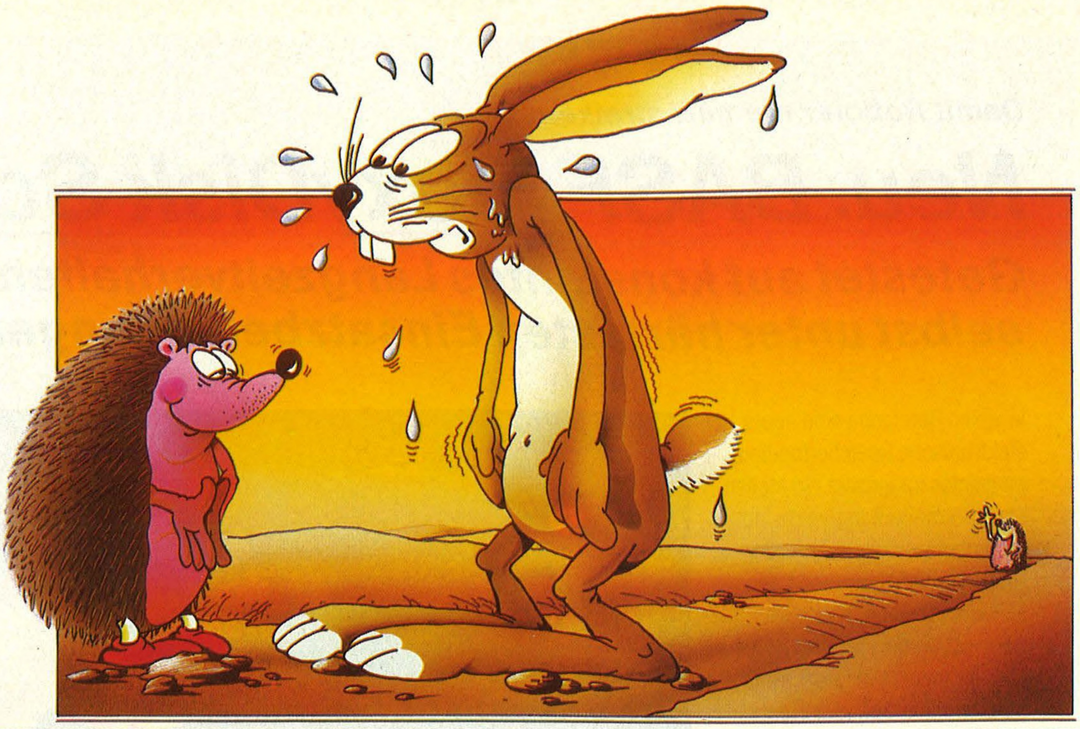
Was wir uns gewünscht hätten wäre, die Version auch auf einem anderen Speicher-Medium als auf dem verhältnismäßig langsamen und umständlichen Band angeboten zu bekommen.

Hersteller: Acornsoft

Geeignet: für BBC Microcomputer Model B und Electron

Preis: circa 60 Mark

Scan: S. Höltgen



Hase und Igel

Salat und Karotten sind die Zahlungsmittel, die über Erfolg oder Mißerfolg bei diesem Familienspiel für bis zu sechs Teilnehmer (Personen oder Computer) entscheiden. Der TI-99/4A mit Speichererweiterung, Extended-BASIC und Joystick überwacht die Regeln und führt die Konten

Jeder Spieler hat 1 Spielstein und anfangs 98 Karotten und 3 Salate. Ziel des Spiels ist es, möglichst als erster das Zielfeld zu erreichen. Wenn man an der Reihe ist, darf man entweder vorwärts gehen, stehen bleiben oder rückwärts gehen. Dabei darf immer nur ein Stein auf einem Feld sein.

Die Regeln im einzelnen

Vorwärts gehen

Pro Feld muß man mit Karotten zahlen, und zwar 1 Karotte für das 1. Feld, 1+2 Karotten für 2 Felder, 1+2+3 Karotten für 3 Felder usw. Es gilt die Formel $n*(n+1)/2$, wobei n die Anzahl der Felder ist, die man in einem Zug geht. Das muß man aber nicht lernen, weil der Computer die Kosten für jedes gegangene Feld sofort anzeigt. Beim Vorwärtsgehen ist darauf zu achten, daß Igelfelder übersprungen werden müssen.

Stehen bleiben

Das darf man nur auf Feldern, die 2 Karotten zeigen. Wenn man stehen bleiben will, muß man entweder 10 Karotten nehmen oder 10 Karotten abgeben.

Rückwärts gehen

Dies kann man nur auf das nächstliegende Feld mit einem Igel, wenn es frei ist. Pro gegangenem Feld erhält man 10 Karotten. Der Computer zeigt die Summe sofort an.

Die Salate

Man muß sie gefressen haben, ehe man ins Ziel läuft. Fressen darf man aber nur auf Feldern, die einen Salatkopf zeigen. Wenn man ein Salatfeld erreicht hat, bleibt man stehen. In der nächsten Runde geht man nicht weiter, sondern frißt 1 Salat, wofür man 10mal soviel Karotten erhält, wie der gerade aktuellen Position der Spielsteine entspricht.

Ist man z. B. beim Fressen an 3. Stelle, bekommt man $10*3=30$ Karotten. Auch diesen Bonus zeigt der Computer an. Wenn man keinen Salat mehr hat, darf man Salatfelder nicht mehr betreten.

Prämienfelder

Alle Felder, die eine Zahl zeigen, sind Prämienfelder. Wenn man wieder an der Reihe ist und die Ziffer des Feldes mit der Position des Steins im Rennen übereinstimmt, bekommt man 10mal soviel Karotten wie der Position entspricht; z. B. an 5. Stelle auf einem Zahlenfeld 1, 5, 6 bekommt man $10*5=50$ Karotten. Der Computer zeigt Prämien bei Zugbeginn automatisch an und addiert sie zum Karottenkonto des Spielers.

Hasenfelder

Bei Feldern, auf der ein Hase abgebildet ist, kommt der Zufall ins Spiel. Wenn der Stein auf einem solchen Feld bleibt, muß man eine Hasenkarte ziehen. Es gibt 8 verschiedene Bestimmungen auf insgesamt 12 Hasenkarten. Die Bestimmungen „10 Karotten geben oder nehmen", „Zug ist kostenlos" und „Um 1 Position zurück" gibt es zweimal, alle anderen nur einmal.

Der Computer mischt die Hasenkarten, wenn alle 12 einmal gezogen worden sind. Das Mischen wird angezeigt.

Die Bestimmung der Hasenkarte wird auf dem Bildschirm ausgedruckt. Nachdem man den Feuerknopf des Joystick gedrückt hat, wird sie automatisch ausgeführt.

Zieleinlauf

Hat man am Anfang für seine Züge oft zuwenig Karotten, so hat man in Zielnähe oft zuviel davon. Es gilt nämlich die Regel, daß der erste nur 10 Karotten, der zweite nur 20, der dritte nur 30 Karotten usw. übrig haben darf, wenn er den letzten Zug bezahlt hat.

Programmablauf

Im unteren Teil des Titelbildes gibt man die Namen der 6 Mitspieler ein. Tippt man

Scan: S. Höltgen

ein „C", so spielt der Computer mit diesem Stein. Das Bild wird gelöscht und das Spielfeld erscheint. Der Start ist vor dem ersten Feld links oben, das Ziel nach dem letzten Feld rechts unten. Auf einer weißen Fläche unterhalb des Spielfeldes erscheinen die Informationen beim Spielablauf.

Es wird der jeweilige Spieler aufgerufen, sein Salat- und Karottenkonto angezeigt, evtl. Prämien usw. Der Spielstein wird mit dem Joystick bewegt, und zwar nach rechts gedrückt (vorwärts) oder nach links gedrückt (rückwärts). Der Stein hüpft mit einem Piepton von Feld zu Feld, neben dem Karottenkonto werden die jeweils verbrauchten oder erhaltenen Karotten angezeigt.

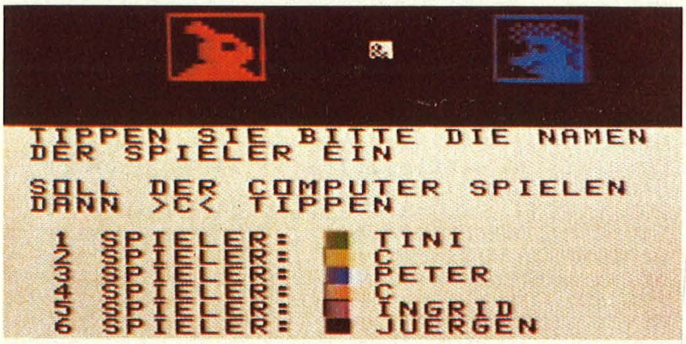
Man beendet den Zug mit einem Druck auf den Feuerknopf. Ist das Feld erlaubt, so bleibt der Stein dort stehen und ist nun unbeweglich. Solange man aber den Feuerknopf nicht gedrückt hat, kann man vorwärts und rückwärts gehen und sich noch anders entscheiden. Ist man schon auf einem Karottenfeld und will nicht ziehen, muß man nur den Feuerknopf drücken.

Der Computer läßt nur erlaubte Züge zu. Wenn die Karotten für den weiteren Schritt nicht mehr reichen, hüpft der Stein auch nicht weiter, ebenso kann man nicht auf den Igel zurück, wenn dieser besetzt ist.

Will man auf verbotenen Feldern stehen bleiben, erscheint mit Ton die Schrift: „NICHT ERLAUBT". Beim Zug vom Ausgangsfeld zum Igel werden automatisch alle Felder dazwischen übersprungen, ebenso vom Igel zum Ausgangsfeld, wenn man den Igelzug rückgängig machen will.

Wenn der Computer spielt, erscheint nach einer kurzen „Denkpause" die Schrift: "FEUERKNOPF". Erst wenn man ihn drückt, wird der Computerzug ausgeführt. Man hat so immer die Kontrolle darüber, wie der Computer gezogen hat. Da bei der Positionsberechnung auch immer alle Steine mitzählen, die das Ziel schon erreicht haben, werden sie unter der letzten Spielfeldreihe neben der Schrift "ZIEL –>" angezeigt. Wenn alle Steine das Ziel erreicht haben, erscheint eine Siegerliste.

Bernhard Decker



A,B	Koordinaten des Spielsteins am Bildschirm
A$	Stringzwischenspeicher
C()	Farbcodespeicher
CL()	Farbcodespeicher der Steine im Ziel
CH()	Erste Charakternummer eines Spielfeldes
E	Entfernung des Spielsteins zum Ziel
F()	Spielfeld
G(X)	Karottenkosten für X Schritte
G$()	Spielernamen der Sieger
H()	Code der gemischten Hasenkarten
HA()	Codespeicher der Hasenkarten
HA$()	Text der Hasenkarten
I,J,K,Q,R	Zählvariable in Schleifen u. ä.
L	Ziffer der nächsten Hasenkarte
M()	Karottenkonto der Spieler
N	Anzahl Schritte pro Zug oder Flag
O	Zwischenspeicher für E-N
P	Position im Rennen
PZ()	Spielfeldnummer der Spielsteine
S()	Salatkonto der Spieler
S$()	Spielernamen
T	Flag, zeigt an, ob „FRISS-SALAT"-Hasenkarte noch kommt oder nicht
V	Zwischenspeicher für PZ()
W	Mögliche Zugweite eines Steins
X,Y	Zwischenspeicher
Z	Zieleinlaufnummer
Z$	Speicher für „ZIEL–>"

Variablenliste

10 bis	60	Impressum
70 bis	250	Initialisierung
85 bis	100	Data Spielfeld (1, 2, 3, 4 : Prämienfelder, 5: Karottenfeld, 6: Hasenfeld, 7: Salat, 8: Igel, 0: Start, 9: Ziel)
	105	Data für erste Charakternr. eines Spielfeldes

110 bis	115	Data-Text Hasenkarten
	120	Data Farbcode
	125	Data-Code Hasenkarten
130 bis	140	Scheinbefehle für Prescanning
	145	Prescanning abschalten
	150	Umdefinition einzelner Charakter
	155	Titelbild
220 bis	245	Abfrage Spielernamen
260 bis	375	Spielerschleife
	270	nächster Spieler
	290	Angabe Salat- u. Karottenkonto (diese Zeile löschen, wenn man verdeckt spielen will)
295 bis	315	Test auf Aussetzen, Salat-, Positionsfeld, zugfähig
	320	Ist die Schrittzahl n=–1, dann neuer Zug
345 bis	370	Zieleinlauf
385 bis	430	Vorwärtszug: Update Karottenkonto, Test auf Hasenfeld, besetztes Feld (Feldwert+50), besetztes Salatfeld (Feldwert+800)
440 bis	855	Computerzug
445 bis	660	Taktik, wenn Computer noch Salat hat. Primär Suche nach Salatfeld, dann 4 Taktiken, d. h. Prioritätslisten, nach denen das Endfeld ausgewählt wird. 670 bis 855 Taktik, wenn Computer kein Salat hat
	670	Wenn Ziel erreichbar, dann setzen
690 bis	790	Taktik, wenn Computer zuwenig Karotten hat
690 bis	715	Test, ob Ziel in 2 Zügen erreichbar
	720	Wenn möglich, erst kleinen Schritt, sonst
	725	erst großen Schritt setzen
730 bis	790	Möglichkeiten, Karotten zu bekommen oder sparen
800 bis	855	Taktik bei zuviel Karotten. Je nach Feld und Überschußhöhe Karottenfeld anpeilen oder auf Igel gehen oder Minischritt. Prämienfelder meiden

Scan: S. Höltgen

865 bis 905	Siegerliste
910 bis 955	Zugausf.: "Auf Karottenfeld bleiben und nehmen"
965	Computerzug vorwärts ausführen
970	Computerzug rückwärts ausführen
980 bis 1000	Position des Steins bestimmen
1010 bis 1040	Test: Stein auf Positionsfeld? Wenn ja, Prämie
1045 bis 1070	Mögliche Zugweite ermitteln
1080 bis 1110	Test: Igelfeld frei?
1120 bis 1220	Nächstmögliche vordere (hintere) Position suchen
1230 bis 1275	Nächstmögliches vordere (hintere) Karottenfeld suchen
1285 bis 1410	Auswertung der Hasenkarten
1420 bis 1430	Auswertung Hasenkarte 6 für Computer
1440 bis 1465	Freies Hasenfeld suchen
1475 bis 1505	Freies Positionsfeld mit richtiger Position suchen
1515 bis 1535	Kleinstmöglichen Schritt gehen
1545 bis 1585	Freies Karottenfeld von nah (fern) suchen
1595 bis 1615	Programmstop, bis Feuerknopf gedrückt wird
1625 bis 1670	Test, ob Stein zugfähig ist
1685 bis 1760	Sonderzeichen definieren
1770 bis 1820	Spielfeld malen
1830 bis 1860	Hasenkarten mischen und Flag T=0 setzen
1865 bis 1930	Bildschirmkoordinaten des Spielsteins berechnen
1940 bis 2085	Spielerzug mit Joystick und Feuerknopf
2095 bis 2205	Routine für graphische Steinbewegung
2215 bis 2225	„NICHT ERLAUBT"-Anzeige
2235 bis 2255	Computerstein blinken lassen
2260 bis 2270	Igelzug ausführen

Programmbeschreibung

```
10 !* HASE & IGEL
15 !* SPIEL V. DAVID PARLETT
20 !* PROGRAMM VERS. 3 FUER
25 !* TI 99/4(A) - KONSOLE
30 !* + MEMORY EXPANSION +
35 !* JOYSTICK + EXT. BASIC
40 !* COPYRIGHT 5/1984 BY
45 !* BERNHARD DECKER
50 !* IM KLOSTERFELD 35
55 !* 8044 UNTERSCHLEISSHEIM
60 !* TEL.- 089/3103631
65 !************************
70 DIM S$(6),M(6),PZ(6),F(65),S(6),G$(6)
,CL(6)
75 DIM CH(9),HA$(9),HA(12),H(12),C(6)
80 DEF G(X)=X*(X+1)/2
85 DATA 0,6,5,6,3,5,6,7,8,4,2,8,3,5,6,8,
1,2,4,8,3,5,7
90 GOTO 145
95 DATA 2,8,6,5,4,3,2,8,6,1,5,6,2,3,8,5,
6,5,2,7
100 DATA 8,3,4,6,2,1,5,8,6,3,2,4,5,8,7,6
,5,2,6,7,6,9
105 DATA 40,96,108,116,100,104,44,112
110 DATA NAECHSTE RUNDE AUSSETZEN,ZUM NA
ECHSTEN KAROTTENFELD,ZUM LETZTEN KAROTTE
NFELD,ZIEHE NOCH EINMAL,ZUG IST KOSTENLO
S
115 DATA 10 KAROTTEN NEHMEN$GEBEN,FRISS
SOFORT 1 SALAT,UM 1 POSITION VOR,UM 1 PO
SITION ZURUECK
120 DATA 13,11,5,10,14,2
125 DATA 1,2,3,4,5,5,6,6,7,8,9,9
130 I,N,Q,R,X,Y,Z,K,L,V,W,E,J,P,A,B,T,O,
A$,Z$
135 CALL CLEAR :: CALL SCREEN :: CALL CH
AR :: CALL CHARPAT :: CALL HCHAR :: CALL
 KEY :: CALL SPRITE :: CALL DELSPRITE
140 CALL LOCATE :: CALL MAGNIFY :: CALL
COLOR :: CALL SOUND :: CALL JOYST
145 !@P-
150 CALL CHARPAT(45,A$):: CALL CHAR(36,A
$):: CALL CHARPAT(43,A$):: CALL CHAR(33,
A$):: CALL CHARPAT(46,A$):: CALL CHAR(37
,A$)
155 CALL CLEAR :: CALL SCREEN(2):: GOSUB
 1685 :: CALL HCHAR(1,1,39,768)
160 CALL MAGNIFY(4):: CALL SPRITE(#1,104
,7,20,64,#2,112,5,20,170):: CALL HCHAR(5
,17,38)
165 IMAGE !###
170 IMAGE $###
175 IMAGE =###
180 Z=1 :: L=13 :: Z$="ZIEL=>"
185 FOR I=1 TO 6
190 S(I)=3 :: M(I)=98 :: PZ(I)=0
195 NEXT I
200 FOR I=0 TO 64 :: READ F(I):: NEXT I
205 FOR I=1 TO 8 :: READ CH(I):: NEXT I
210 FOR I=1 TO 9 :: READ HA$(I):: NEXT I
215 FOR I=1 TO 6 :: READ C(I):: NEXT I
220 CALL HCHAR(10,1,32,448):: DISPLAY AT
(10,1):"TIPPEN SIE BITTE DIE NAMEN  DER
SPIELER EIN"
225 DISPLAY AT(13,1):"SOLL DER COMPUTER
SPIELEN   DANN >C< TIPPEN"
230 FOR I=1 TO 6
235 DISPLAY AT(15+I,1):I;"SPIELER: ";CHR
$(116+4*I)
240 ACCEPT AT(15+I,15):S$(I)
245 NEXT I
250 RANDOMIZE :: CALL DELSPRITE(ALL):: C
ALL MAGNIFY(3):: GOSUB 1770
255 !SPIELERSCHLEIFE
260 FOR I=1 TO 6
265 CALL HCHAR(16,1,32,288):: CALL HCHAR
(1,1,39,6):: CALL HCHAR(2,1,39,6)
270 IF PZ(I)=64 THEN 375
275 IF S$(I)<>"C" THEN DISPLAY AT(16,1):
"ZUG VON ";CHR$(116+4*I);" ";S$(I):: GOT
O 285
280 DISPLAY AT(16,1):"SPIELER";I;CHR$(11
6+4*I);" COMPUTERZUG"
285 V=PZ(I):: E=64-V :: GOSUB 1010
290 CALL SPRITE(#3,44,4,130,17,#4,100,10
,130,97):: DISPLAY AT(18,3)SIZE(3):"=";S
(I):: DISPLAY AT(18,13):USING 175:M(I)
295 IF F(V)>1000 THEN DISPLAY AT(21,9):"
SETZT AUS" :: F(V)=F(V)-950 :: GOTO 345
300 IF F(V)>800 THEN DISPLAY AT(22,5):"F
RISST" :: CALL SPRITE(#5,44,4,161,97)ELS
E 315
305 DISPLAY AT(18,18):"!";10*P
310 M(I)=M(I)+10*P :: F(V)=F(V)-750 :: S
(I)=S(I)-1 :: GOTO 345
315 GOSUB 1050 :: GOSUB 1625
320 IF N=-1 THEN 275
325 IF S$(I)="C" THEN 440 ELSE DISPLAY A
T(24,10)BEEP:"JOYSTICK" :: GOTO 1940
330 PZ(I)=V
335 J=116+4*I :: CALL HCHAR(A,B,J):: CAL
L HCHAR(A+1,B,J+1):: CALL HCHAR(A,B+1,J+
2):: CALL HCHAR(A+1,B+1,J+3)
340 IF N=-1 THEN 265
345 IF V=64 THEN F(V)=9 :: Z=Z+1 ELSE 37
0
350 CALL SOUND(800,880,4,1109,4,1319,3)
355 DISPLAY AT(21,1):"SIE SIND IM ZIEL U
ND": :STR$(Z-1);"% SIEGER"
360 G$(Z-1)=S$(I):: CL(Z-1)=I
365 Z$=Z$&" "&CHR$(116+4*I):: DISPLAY AT
(15,29-LEN(Z$))SIZE(LEN(Z$)):Z$ :: CALL
HCHAR(13,29,39,2):: CALL HCHAR(14,29,39,
2)
370 GOSUB 1595 :: CALL DELSPRITE(ALL)::
IF Z=7 THEN 865
375 NEXT I :: GOTO 260
380 !ZUG VORWAERTS
385 E=64-V :: M(I)=M(I)-G(N)
390 IF F(V)<>6 THEN 425
395 CALL SOUND(200,800,4):: CALL SOUND(4
00,600,7)
400 CALL HCHAR(21,1,32,128):: CALL SPRIT
E(#2,104,7,160,16)
405 IF L=13 THEN DISPLAY AT(22,3):"MISCH
T" :: GOSUB 1830
410 J=H(L):: DISPLAY AT(22,3):HA$(J):: L
=L+1 :: GOSUB 1595
415 CALL HCHAR(21,1,32,128):: GOSUB 1285
 :: CALL DELSPRITE(#2)
420 IF N<-1 THEN N=N+2 ELSE 390
425 IF F(V)=7 THEN F(V)=807 ELSE F(V)=F(
V)+50
430 RETURN
435 !COMPUTERZUG
440 GOSUB 1870 :: IF S(I)=0 THEN 670
445 IF V<24 OR S(I)=1 THEN 465
450 FOR N=MIN(W,57-V)TO 1 STEP -1 !S-FLD
 WEIT
455 IF F(V+N)=7 THEN 965
```

Scan: S. Höltgen


```
460 NEXT N
465 FOR N=1 TO MIN(62-V,W)
470 IF F(V+N)=7 THEN 965
475 NEXT N
480 IF V<7 AND(PZ(MAX(I-1,1))=7 OR P=6)T
HEN 485 ELSE 500
485 IF F(V)-50=5 THEN 915 ELSE GOSUB 154
5
490 IF N<0 OR N>4 THEN GOSUB 1515
495 GOTO 965
500 IF T=0 THEN 510 ELSE GOSUB 1440
505 IF N>0 THEN 965
510 IF M(I)+S(I)*10*Z>G(E)/(S(I)+1)THEN
545
515 IF V<11 OR V>24 THEN 555
520 IF V>22 AND S(I)<2 THEN 555
525 IF V>22 THEN 545
530 IF (PZ(MAX(I-1,1))=22 AND PZ(I)<22)O
R P=6 THEN 535 ELSE 555
535 IF PZ(I)<16 THEN GOSUB 1475 ELSE 545
540 IF N>-1 THEN 965
545 GOSUB 1080 :: IF N>0 THEN 970
550 GOSUB 1515 :: GOTO 965
555 IF (V<47 AND V>37 AND S(I)>1)OR(V>42
 AND M(I)>G(E))THEN GOSUB 1080 ELSE 565
560 IF N>0 THEN 970
565 ON INT(RND*4)+1 GOTO 575,605,625,655
570 !TAKTIK HASE,KAROTTENFELD,POSITION,I
GEL,BLEIBEN,KL.SCHRITT
575 GOSUB 1440 :: IF N>-1 THEN 965 ELSE
GOSUB 1545
580 IF N>0 THEN 965 ELSE GOSUB 1475
585 IF N>0 THEN 965 ELSE GOSUB 1080
590 IF N>0 THEN 970
595 IF F(V)-50=5 THEN 915 ELSE GOSUB 151
5 :: GOTO 965
600 !TAKTIK POSITION,IGEL,BLEIBEN,KL.SCH
RITT
605 GOSUB 1475 :: IF N>0 THEN 965 ELSE G
OSUB 1080
610 IF N>0 THEN 970
615 IF F(V)-50=5 THEN 915 ELSE GOSUB 151
5 :: GOTO 965
620 !TAKTIK KAROTTENFELD,HASE,POSITION,B
LEIBEN,IGEL,KL.SCHRITT
625 GOSUB 1545 :: IF N>0 THEN 965 ELSE G
OSUB 1440
630 IF N>0 THEN 965 ELSE GOSUB 1475
635 IF N>0 THEN 965 ELSE IF F(V)-50=5 TH
EN 915
640 GOSUB 1080 :: IF N>0 THEN 970
645 GOSUB 1515 :: GOTO 965
650 !TAKTIK POSITION,KL.SCHRITT,IGEL
655 GOSUB 1475 :: IF N>0 THEN 965 ELSE G
OSUB 1515
660 IF N>0 THEN 965 ELSE GOSUB 1080 :: G
OTO 970
665 !COMPT.ZUG OHNE SALAT
670 IF M(I)<=G(E)+10*Z AND W>=E THEN N=E
 :: GOTO 965
675 IF W>E THEN 800
680 !ZUWENIG KAROTTEN
685 !ZIEL IN 2 ZUEGEN?
690 FOR N=1 TO INT(W/2)+1
695 O=E-N
700 IF M(I)-G(N)-G(O)>10*Z THEN 730
705 IF G(N)+G(O)<=M(I)AND M(I)-G(N)-G(O)
<=10*Z THEN 720
710 NEXT N
715 GOTO 735
720 IF F(V+N)>6 THEN N=0 ELSE 965
725 IF F(V+N)>6 THEN 735 ELSE 965
730 GOSUB 1080 :: IF N>0 THEN 970 ELSE 7
45
735 IF V<51 THEN 775
740 GOSUB 1570 :: IF N>0 THEN 965
745 IF M(I)-G(E)<=40 THEN IF F(V)-50=5 T
HEN 915
750 N=E-W
755 IF F(V+N)>6 THEN N=N+1
760 IF N<=W THEN 965 ELSE GOSUB 1515
765 IF N>0 THEN 965 ELSE GOSUB 1080
770 IF N>0 THEN 970 ELSE 915
775 GOSUB 1475 :: IF N>0 THEN 965 ELSE G
OSUB 1570
780 IF N>0 THEN 965 ELSE 745
785 IF F(V+N)<7 THEN 965 ELSE GOSUB 1515
790 IF N>0 THEN 965 ELSE 765
795 !ZUVIEL KAROTTEN
800 IF V<53 OR M(I)>G(E)+10*Z+40 THEN 82
0
805 IF F(V)-50=5 THEN 915 ELSE GOSUB 154
5
810 IF N>0 THEN 965 ELSE GOSUB 1515
815 IF N>0 THEN 965
820 GOSUB 1080 :: IF N>0 THEN 970
825 IF F(V)-50=5 THEN 915 ELSE N=1
830 IF F(V+N)>10 THEN P=P-1
835 IF F(V+N)=1 AND P>4 THEN 845
840 IF F(V+N)>6 OR F(V+N)=P THEN 845 ELS
E 965
845 N=N+1
850 IF N<=W THEN 830 ELSE GOSUB 1515
855 GOTO 965
860 !ENDE
865 CALL HCHAR(16,1,32,288)
870 DISPLAY AT(16,1):"ERGEBNIS:"
875 FOR I=1 TO 6
880 DISPLAY AT(16+I,4):STR$(I);"% SIEGER
: ";CHR$(116+4*CL(I))
885 IF G$(I)="C" THEN G$(I)="COMPUTER"
890 DISPLAY AT(16+I,17):G$(I)
895 NEXT I
900 GOSUB 1595
905 STOP
910 !KAROTTENFELD BLEIBEN U NEHMEN
915 N=0
920 DISPLAY AT(21,3):"BLEIBT STEHEN"
925 IF M(I)+S(I)*Z*10>G(E)/(S(I)+1)THEN
945
930 DISPLAY AT(18,18):"! 10"
935 M(I)=M(I)+10
940 GOTO 330
945 DISPLAY AT(18,18):"$ 10"
950 M(I)=M(I)-10
955 GOTO 330
960 !COMPT.ZUG AUSFUEHREN
965 GOSUB 2235 :: K=1 :: GOSUB 2265 :: C
ALL COLOR(#1,C(I)):: DISPLAY AT(18,18):U
SING 170:G(N):: GOTO 2075 !VOR
970 GOSUB 2235 :: K=-1 :: GOSUB 2260 ::
CALL COLOR(#1,C(I)):: GOTO 2075 !RUECK
975 !POSITION BESTIMMEN
980 P=1
985 FOR X=1 TO 6
990 IF PZ(X)>V THEN P=P+1
995 NEXT X
1000 RETURN
1005 !STEIN AUF POSITIONSFELD?
1010 GOSUB 980
1015 IF (F(V)-50=P AND P<5)OR(F(V)-50=1
AND P>4)THEN 1025
1020 RETURN
1025 CALL SOUND(1000,659,4,831,4,988,3)
1030 FOR J=1 TO 7 :: DISPLAY AT(18,1):"P
RAEMIE =";10*P;"KAROTTEN" :: CALL HCHAR(
18,1,32,32):: NEXT J
1035 M(I)=M(I)+10*P
1040 RETURN
1045 !MAX ZUGWEITE
1050 X=M(I):: W=1
1055 IF X-W<0 THEN 1065
1060 X=X-W :: W=W+1 :: GOTO 1055
1065 W=W-1
1070 RETURN
1075 !IGEL FREI?
1080 FOR N=1 TO 8
1085 IF V-N<=0 THEN N=-2 :: RETURN
1090 IF F(V-N)>50 THEN IF F(V-N)-50=8 TH
EN 1105
1095 IF F(V-N)=8 THEN 1110
1100 NEXT N
1105 N=-1
1110 RETURN
1115 !VORDERE POSITION SUCHEN
1120 GOSUB 980
1125 IF P=1 THEN 1245
1130 FOR X=1 TO 64-V
1135 IF F(V+X)>50 THEN 1150
1140 NEXT X
1145 X=X-1
1150 X=X+1
1155 IF V+X=64 AND(S(I)>0 OR M(I)-G(X)>1
0*Z)THEN 1245
1160 IF F(V+X)>50 OR(F(V+X)=7 AND S(I)=0
)OR F(V+X)=8 THEN 1150
1165 DISPLAY AT(21,3):X;"FELDER VORWAERT
S" :: K=1 :: V=MIN(64,V+X)
```

Scan: S. Höltgen


```
1170 RETURN
1175 !HINTERE POSITION SUCHEN
1180 GOSUB 980 :: IF P=6 THEN 1245
1185 FOR X=1 TO V
1190 IF F(V-X)>50 THEN 1205
1195 NEXT X
1200 X=X-1
1205 X=MIN(V,X+1)
1210 IF F(V-X)>50 OR(F(V-X)=7 AND S(I)=0
)THEN 1205
1215 DISPLAY AT(21,3):X;"FELDER ZURUECK"
:: K=-1 :: V=MAX(0,V-X)
1220 RETURN
1225 !VORDERES KAROTTENFELD SUCHEN
1230 FOR X=1 TO 60-V
1235 IF F(V+X)=5 THEN 1165
1240 NEXT X
1245 DISPLAY AT(21,3):"WIRKUNGSLOS" :: K
,X=0 :: N=-2
1250 RETURN
1255 !HINTERES KAROTTENFELD SUCHEN
1260 FOR X=1 TO V-2
1265 IF F(V-X)=5 THEN 1215
1270 NEXT X
1275 GOTO 1245
1280 !AUSWERTUNG HASENKARTE SPIELER
1285 K,X=0 :: IF S$(I)="C" AND J=6 THEN
GOSUB 1420 :: GOTO 1345
1290 ON J GOSUB 1305,1310,1315,1320,1325
,1330,1335,1350,1180
1295 IF K<>0 THEN N=X :: GOSUB 2265
1300 RETURN
1305 F(V)=F(V)+950 :: GOTO 1345 !1
1310 GOSUB 1230 :: GOTO 1345 !2
1315 GOSUB 1260 :: GOTO 1345 !3
1320 N=-3 :: RETURN !4
1325 M(I)=M(I)+G(N):: DISPLAY AT(18,23):
USING 165:G(N):: GOTO 1345 !5
1330 GOSUB 1360 :: GOTO 1345 !6
1335 IF S(I)>0 THEN T=0 :: F(V)=F(V)+750
:: GOTO 1345 !7
1340 F(V)=F(V)+950
1345 N=-2 :: RETURN
1350 GOSUB 1120 :: RETURN !8
1355 !KAROTTEN NEHMEN?
1360 DISPLAY AT(20,3):"WOLLEN SIE"
1365 DISPLAY AT(21,3):"A: 10 KAROTTEN NE
HMEN":"  B: 10 KAROTTEN ABGEBEN"
1370 IF J<>6 THEN 1380
1375 DISPLAY AT(23,3):"C: NICHTS TUN?"
1380 CALL KEY(0,Q,R)
1385 IF R=0 OR Q<65 OR Q>67 THEN 1380
1390 IF Q=67 THEN 1410
1395 IF Q=65 THEN DISPLAY AT(18,23):"! 1
0 " :: M(I)=M(I)+10 :: GOTO 1410
1400 IF M(I)>=10 THEN DISPLAY AT(18,23):
"$ 10" :: M(I)=M(I)-10 :: GOTO 1410
1405 GOSUB 2215 :: N=-1
1410 RETURN
1415 !HASENKARTE WERTEN COMP
1420 IF M(I)>G(E)+10*Z THEN Q=66 :: GOTO
 1400
1425 IF M(I)<G(E)THEN Q=65 ELSE Q=67
1430 GOTO 1390
1435 !FREIES HASENFELD SUCHEN
1440 FOR N=1 TO MIN(W,7)
1445 IF V+N=64 THEN 1460
1450 IF F(V+N)=6 THEN 1465
1455 NEXT N
1460 N=-1
1465 RETURN
1470 !POSITIONSFELD SUCHEN
1475 GOSUB 980 :: FOR N=1 TO MIN(60-V,W)
1480 IF F(V+N)>50 THEN P=P-1 :: GOTO 149
5
1485 IF (F(V+N)=P AND P<5)OR(F(V+N)=1 AN
D P>4)THEN 1505
1490 IF F(V+N)=1 THEN IF P>4 THEN 1505
1495 NEXT N
1500 N=-1
1505 RETURN
1510 !MINI-SCHRITT
1515 N=1
1520 IF N>W OR V+N>64 THEN 1460
1525 IF V+N=64 AND(S(I)>0 OR M(I)-G(N)>1
0*Z)THEN 1460
1530 IF (F(V+N)=7 AND S(I)=0)OR F(V+N)>7
 THEN N=N+1 :: GOTO 1520
1535 RETURN
1540 !FREIES KAROTTENFELD SUCHEN (NAH)
1545 FOR N=1 TO MIN(59-V,W)
1550 IF F(V+N)=5 THEN 1560
1555 NEXT N :: N=-1
1560 RETURN
1565 !FREIES KAROTTENFELD SUCHEN (WEIT)
1570 FOR N=MIN(W,59-V)TO 1 STEP -1
1575 IF F(V+N)=5 THEN 1585
1580 NEXT N :: N=-1
1585 RETURN
1590 !WEITERSCHALTEN
1595 CALL SOUND(50,800,7)
1600 DISPLAY AT(24,18):"FEUERKNOPF"
1605 CALL KEY(1,Q,R):: IF R=0 THEN 1605
1610 CALL HCHAR(24,1,32,32)
1615 RETURN
1620 !ZUGFAEHIG?
1625 IF F(V)-50=5 THEN 1675
1630 GOSUB 1080 :: IF N>0 THEN 1675
1635 FOR N=1 TO MIN(64-V,W)
1640 IF F(V+N)<7 OR(F(V+N)=7 AND S(I)>0)
THEN 1675
1645 IF F(V+N)=9 AND(M(I)-G(N)<=10*Z AND
 S(I)=0)THEN 1675
1650 NEXT N
1655 DISPLAY AT(20,1):"SIE KOENNEN NICHT
 ZIEHEN": :"SIE ERHALTEN 98 KAROTTEN":"U
ND BEGINNEN VON VORNE"
1660 N=-1 :: M(I)=98 :: GOSUB 1595 :: GO
SUB 1870 :: CALL HCHAR(20,1,32,160):: PZ
(I)=0 :: F(V)=MAX(0,F(V)-50)
1665 CALL HCHAR(A,B,CH(F(V))):: CALL HCH
AR(A+1,B,CH(F(V))+1):: CALL HCHAR(A,B+1,
CH(F(V))+2):: CALL HCHAR(A+1,B+1,CH(F(V)
)+3)
1670 RETURN
1675 N=0 :: RETURN
1680 !SONDERZEICHEN
1685 CALL CHAR(39,"FFFFFFFFFFFFFFFF")
1690 CALL CHAR(40,"FF8080818280B8A0A0B88
484A49880FFFF0181818199A1A1B9252525190
1FF")!=1
1695 CALL CHAR(44,"FF80808083878580D8F8F8
C8783808OFFFF010101C1B169E959F9F9F1E1010
1FF")!SALAT
1700 CALL CHAR(96,"FF80808387848080818 38
687878080FFFF0101C1E16161C1810101E1E1010
1FF")!=2
1705 CALL CHAR(100,"FF8080808080818387 8E
8C81838380FFFF05193971E7D9B93171E1C18101
01FF")!MOEHREN
1710 CALL CHAR(104,"FF808C9E9F8783878787
87838F9FBFFFFF01010101E1F1999DFDFDF9F1F9
FDFF")!HASE
1715 CALL CHAR(112,"FF808A958A978B828387
9F9E858380FFFF01A149B5E9F57B7DFBFBFBFDFF
FFFF")!IGEL
1720 CALL CHAR(108,"FF808083878480818180
8487838080FFFF0101C1E16161C1E16161E1C101
01FF")!=3
1725 CALL CHAR(116,"FF808080818386 8C8C8F
8F80808080FFFF0101C1C1C1C1C1F1F1C1C101
01FF")!=4
1730 A$="FF80808080808080" :: CALL CHAR(
120,A$,128,A$,136,A$):: A$="007F7F7F7F7F
7F7F" :: CALL CHAR(124,A$,132,A$,140,A$)
1735 A$="80808080808080FF" :: CALL CHAR(
121,A$,129,A$,137,A$):: A$="7F7F7F7F7F7F
7F00" :: CALL CHAR(125,A$,133,A$,141,A$)
1740 A$="FF01010101010101" :: CALL CHAR(
122,A$,130,A$,138,A$):: A$="00FEFEFEFEFE
FEFE" :: CALL CHAR(126,A$,134,A$,142,A$)
1745 A$="01010101010101FF" :: CALL CHAR(
123,A$,131,A$,139,A$):: A$="FEFEFEFEFEFE
FE00" :: CALL CHAR(127,A$,135,A$,143,A$)
1750 FOR I=0 TO 8 :: CALL COLOR(I,2,16):
: NEXT I
1755 CALL COLOR(2,3,16,9,9,16,10,7,16,11
,5,16,12,11,13,13,10,5,14,2,14)
1760 RETURN
1765 !SPIELFELD MALEN
1770 CALL HCHAR(1,1,39,768)
1775 I=1 :: J=5 :: K=2
1780 FOR N=0 TO 64
1785 CALL HCHAR(I,J,CH(F(N)))
1790 CALL HCHAR(I+1,J,CH(F(N))+1)
1795 CALL HCHAR(I,J+1,CH(F(N))+2)
1800 CALL HCHAR(I+1,J+1,CH(F(N))+3)
1805 J=J+K :: IF J>4 AND J<28 THEN 1820
1810 I=I+1 :: N=N+1 :: CALL HCHAR(I,J,CH
(F(N))):: CALL HCHAR(I+1,J,CH(F(N))+1)::
```

Scan: S. Höltgen


```
 CALL HCHAR(I,J+1,CH(F(N))+2)
1815 CALL HCHAR(I+1,J+1,CH(F(N))+3):: I=
I+2 :: K=-K :: J=J+K
1820 NEXT N :: DISPLAY AT(15,23):Z$ :: R
ETURN
1825 !HASENKARTEN MISCHEN
1830 RESTORE 125
1835 FOR J=1 TO 12 :: READ HA(J):: NEXT
J
1840 FOR J=1 TO 12
1845 X=INT(RND*12)+1
1850 H(J)=HA(X):: IF H(J)=0 THEN 1845 EL
SE HA(X)=0
1855 NEXT J
1860 T,L=1 :: RETURN
1865 !SPIELFELDKOORDINATEN BERECHNEN
1870 IF V<12 THEN A=1 :: B=5+2*V :: GOTO
 1910
1875 IF V=12 THEN A=2 :: B=29 :: GOTO 19
10
1880 IF V<25 THEN A=4 :: B=29-(V-12)*2 :
: GOTO 1910
1885 IF V=25 THEN A=5 :: B=3 :: GOTO 191
0
1890 IF V<38 THEN A=7 :: B=1+(V-24)*2 ::
 GOTO 1910
1895 IF V=38 THEN A=8 :: B=29 :: GOTO 19
10
1900 IF V<51 THEN A=10 :: B=29-(V-38)*2
:: GOTO 1910
1905 IF V=51 THEN A=11 :: B=3 ELSE A=13
:: B=1+(V-50)*2
1910 CALL SPRITE(#1,124,C(I),8*(A-1)+1,8
*(B-1)+1)
1915 IF V=0 THEN 1930
1920 A$=STR$(F(V)):: J=VAL(SEG$(A$,LEN(A
$),1))
1925 CALL HCHAR(A,B,CH(J)):: CALL HCHAR(
A+1,B,CH(J)+1):: CALL HCHAR(A,B+1,CH(J)+
2):: CALL HCHAR(A+1,B+1,CH(J)+3)
1930 RETURN
1935 !SPIELERZUG
1940 GOSUB 1870 :: N=0 :: K=1
1945 CALL JOYST(1,X,Y):: CALL COLOR(#1,1
):: CALL KEY(1,Q,R):: IF X=0 AND R=0 THE
N CALL COLOR(#1,C(I)):: GOTO 1945
1950 CALL HCHAR(24,1,32,32):: GOTO 1960
1955 CALL JOYST(1,X,Y)
1960 CALL COLOR(#1,1):: IF X=0 THEN 2035
1965 IF X<0 AND K=3 THEN 2035
1970 IF X<0 THEN K=-1 ELSE K=1
1975 IF N>0 OR(N=0 AND K=1)THEN 2015
1980 IF N<0 AND K=1 THEN 2005
1985 !ZURUECK AUF IGEL
1990 GOSUB 1080 :: IF N<1 THEN N=0 :: GO
TO 2035
1995 GOSUB 2260 :: K=3 :: GOTO 2035
2000 !IGELZUG ZURUECKNEHMEN
2005 GOSUB 2265 :: N=0 :: CALL HCHAR(18,
20,32,140):: CALL DELSPRITE(#2):: GOTO 2
035
2010 !NORMALSCHRITT
2015 IF N+K>W OR V+N+K>64 OR(V+N+K=64 AN
D(S(I)>0 OR M(I)-G(N+1)>10*Z))THEN 2035
2020 N=N+K :: GOSUB 2095
2025 DISPLAY AT(18,18):USING 170:G(N)
2030 !SPRITE SICHTBAR
2035 CALL COLOR(#1,C(I))
2040 CALL KEY(1,Q,R):: IF R=0 THEN 1955
2045 IF N=0 THEN IF F(V+N)-50=5 THEN GOS
UB 1360 :: GOTO 330 ELSE 2070
2050 IF F(V+N)>10 THEN 2070
2055 IF N>0 AND F(V+N)=8 THEN 2070
2060 IF F(V+N)=7 AND S(I)=0 THEN 2070
2065 GOTO 2075
2070 GOSUB 2215 :: GOTO 1955
2075 F(V)=F(V)-50 :: V=V+N :: IF N>0 THE
N GOSUB 385 :: GOTO 330
2080 IF N<0 THEN M(I)=M(I)-10*N :: F(V)=
F(V)+50 :: N=0 ELSE J=0 :: GOSUB 1360
2085 GOTO 330
2090 !STEIN BEWEGEN
2095 J=INT(A/3)+1 :: IF K<0 THEN 2110
2100 IF J/2=INT(J/2)THEN Y=-2 ELSE Y=2
2105 GOTO 2115
2110 IF J/2=INT(J/2)THEN Y=2 ELSE Y=-2
2115 B=B+Y !NEUE SPALTE
2120 CALL COLOR(#1,C(I))
2125 IF Y<0 THEN IF B>4 AND B<26 THEN 21
95 ELSE 2170
2130 IF B>5 AND B<28 THEN 2195
2135 IF Y<0 THEN 2170
2140 IF A=13 THEN B=MIN(B,29):: GOTO 219
5 !BREMSE ZIEL
2145 IF A=4 OR A=5 OR A=10 OR A=11 THEN
2160
2150 IF A=1 OR A=7 THEN A=A+1 ELSE A=A+2
 :: B=B-Y*2 :: Y=-Y
2155 GOTO 2165
2160 IF A=4 OR A=10 THEN A=A-2 ELSE A=A-
1
2165 GOTO 2195
2170 IF A=1 AND B<3 THEN B=B-Y*2 :: GOTO
 2195 !BREMSE START
2175 IF A=4 OR A=5 OR A=10 OR A=11 THEN
2190
2180 IF A=7 OR A=13 THEN A=A-2 ELSE A=A-
1
2185 GOTO 2195
2190 IF A=4 OR A=10 THEN A=A+1 ELSE A=A+
2 :: B=B-Y*2 :: Y=-Y
2195 CALL COLOR(#1,1):: CALL SOUND(100,3
00,6)
2200 CALL LOCATE(#1,8*(A-1)+1,8*(B-1)+1)
2205 RETURN
2210 !FEHLER
2215 CALL SOUND(700,600,4,620,4,650,4)
2220 FOR J=1 TO 7 :: DISPLAY AT(24,9):"N
ICHT ERLAUBT" :: CALL HCHAR(24,1,32,32):
: NEXT J
2225 RETURN
2230 !COMP STEIN BLINKEN
2235 CALL SOUND(50,800,7)
2240 DISPLAY AT(24,18):"FEUERKNOPF"
2245 CALL KEY(1,Q,R):: CALL COLOR(#1,C(I
)):: CALL COLOR(#1,1):: IF R=0 THEN 2245
2250 CALL HCHAR(24,1,32,32)
2255 RETURN
2260 DISPLAY AT(18,18):"!";N*10: : : :"
      ZUM  ZURUECK" :: N=-N :: CALL SPRI
TE(#2,112,5,161,97)
2265 FOR R=1 TO ABS(N):: GOSUB 2095 :: N
EXT R
2270 RETURN
```

Merkur

Eine meisterhafte Simulation eines Geldspielautomaten mit Start-Stop-Taste, Sonderspielen und Risikoleiste für den Spectrum 48K

Die Eingabe des gesamten Programmes gliedert sich in folgende Schritte:

– Eintippen des Ladeprogramms „Byte-Laden“ und Starten mit RUN.

– Eintippen der Bytes 50000 bis 60704, wobei Eintippfehler durch „v“ (für ein Byte vor) sowie durch „z“ (für eins zurück) behoben werden können. Bei Eingabe von „e“ wird die Eingabe abgeschlossen sowie das Programm durch NEW gelöscht, wobei die eingegebenen Bytes nicht verlorengehen.

– Nun erfolgt die Eingabe des eigentlichen Programms.

– Aufruf der Prüfsummenroutine durch GOTO 2400.

– Wurde während der Byte-Eingabe oder in der DATA-Zeile mit den Prüfsummen ein Eingabefehler gemacht, so erscheint auf dem Bildschirm die Fehlermeldung mit Angabe des Fehlerbereichs der Bytes

Scan: S. Höltgen


```
0;"B"
 400 BEEP .002,t: GO SUB 1000
 450 REM Bewegung der Autos
 500 FOR n=0 TO 1: LET b$=" ": IF RND<rn THEN LET b$="E"
 510 LET a$(6+n*2)=b$+a$(6+n*2, TO 31): NEXT n
 520 LET b$="  ": IF RND<rn THEN LET b$="GH"
 530 LET a$(5)=a$(5,3 TO )+b$
 540 LET b$=" ": IF RND<(rn/2) AND a$(7,31 TO )="  " THEN LET b$="■": LET a$(7,31 TO )="I■"
 550 LET a$(7)=a$(7,2 TO )+b$
 560 IF x<>21 AND x>10 THEN LET q$=a$((x-3)/2,(y+1)): IF q$="■" OR q$="E" OR q$="G" OR q$="H" OR q$="I" THEN GO TO 3000
 562 GO SUB 4000: GO SUB 1000
 565 REM Bewegung der Schildkroeten
 570 FOR n=0 TO 1: LET b$=" ": IF RND<rn1 AND a$(2+n*2,1)=" " THEN LET b$="C": LET a$(2+n*2,1)="D"
 580 LET a$(2+n*2)=b$+a$(2+n*2, TO 31): NEXT n
 585 REM Bewegung der Baumstaemme
 590 FOR n=0 TO 1: LET b$=" ": IF RND<rn2 AND a$(1+n*2,32)=" " THEN LET a$(1+n*2,32)="J": LET b$="K"
 595 IF a$(1+n*2,32)="J" AND fri>0 AND RND<.5 THEN LET a$(1+n*2,32)="Q": LET b$="N"
 600 IF a$(1+n*2,32)="K" OR a$(1+n*2,32)="N" THEN LET b$="K"
 610 IF a$(1+n*2,27)="J" THEN LET b$="L"
 615 IF a$(1+n*2,27)="Q" THEN LET b$="M": LET a$(1+n*2,32)="P"
 620 LET a$(1+n*2)=a$(1+n*2,2 TO )+b$
 630 NEXT n
 640 IF x=3 OR x=7 THEN PRINT AT x,y; PAPER 5;a$(INT (x/2),y+1): LET y=y-1: IF y<0 THEN GO TO 3000
 650 IF x=5 OR x=9 THEN PRINT AT x,y; PAPER 5;a$(INT (x/2),y+1): LET y=y+1: IF y>31 THEN GO TO 3000
 660 GO SUB 4100
 900 GO TO 395
 999 STOP
1000 REM Bewegung des Frosches
1005 LET t=t-tm: PRINT #0;AT 0,t;" ": IF t<=0 THEN GO TO 3000
1010 IF INKEY$<>"5" AND INKEY$<>"6" AND INKEY$<>"7" AND INKEY$<>"8" THEN RETURN
1015 BEEP .002,20: BEEP .002,3
1020 IF x>10 AND x<>21 THEN PRINT AT x,y; PAPER 0;" "
1027 IF x<10 AND x>2 THEN PRINT AT x,y; PAPER 5;a$(INT (x/2),y+1)
1030 IF x=21 THEN PRINT AT x,y; PAPER 7;" "
1035 LET sc=sc+10*((INKEY$="7")-(INKEY$="6" AND x<>21))
1040 LET x=x+2*((INKEY$="6" AND x<>21)-(INKEY$="7")): LET y=y+(INKEY$="8" AND y<>31)-(INKEY$="5" AND y<>0)
1050 IF x=1 AND ATTR (x,y)<>1 AND ATTR (x,y)<>57 THEN GO TO 2000
1055 IF INKEY$="7" OR INKEY$="6" THEN PRINT #0;AT 1,0; INK 7;"SCORE:";FN x$(sc);sc
1060 IF x=21 THEN GO TO 1100
1070 IF ATTR (x,y)=57 OR ATTR (x,y)=1 THEN GO TO 3000
1080 IF x<10 THEN LET q$=a$(INT (x/2),(y+1)): IF q$<>"C" AND q$<>"D" AND q$<>"J" AND q$<>"K" AND q$<>"L" AND q$<>"P" AND q$<>"M" THEN GO TO 3000
1090 IF x>10 THEN LET q$=a$((x-3)/2,(y+1)): IF q$="E" OR q$="G" OR q$="H" OR q$="I" OR q$="■" THEN GO TO 3000
1100 GO SUB 4020
1120 RETURN
2000 REM Frosch im Revier
2003 GO SUB 4000: GO SUB 4100
2004 IF ATTR (1,y-2)<>1 THEN PRINT INK 1;AT 0,y-2;"QR ";AT 1,y-2;"ST ": GO TO 2007
2005 IF ATTR (1,y-1)<>1 THEN PRINT INK 1;AT 0,y-1;"QR ";AT 1,y-1;"ST ": GO TO 2007
2006 PRINT INK 1;AT 0,y;"QR ";AT 1,y;"ST "
2020 RESTORE 2020: FOR n=1 TO 18: READ a,b: BEEP 1/a,b: NEXT n
2030 DATA 3,0,3,5,10,5,2,5,3,0,3,7,10,4,2,5
2040 DATA 3,0,3,5,10,9,3,12,3,9,10,5,3,0,3,7,10,9,2,5
2050 LET goto=260: LET fr=fr+1: IF fr=5 THEN LET fr=0: LET goto=255: LET fri=fri+1
2060 IF fr=0 THEN LET rn=rn+.02*(fr>0): LET tm=tm+.02: LET rn1=rn1-.02*(fri>1): LET rn2=rn2-0.02*(fri>2)
2080 LET t=INT t: FOR n=0 TO t: PRINT #0;AT 0,t;" ": LET t=t-1: BEEP .1,32-t: LET sc=sc+(50*tm): LET sc=INT sc: PRINT #0;AT 1,0;"SCORE:";FN x$(sc);sc: NEXT n: GO TO goto
3000 REM Tod
3001 PRINT #0;AT 1,0;"SCORE:";FN x$(sc);sc;AT 1,14;s$( TO f);" ";AT 1,21;"HSC:";FN x$(hsc);hsc
3002 LET at=ATTR (x,y)
3003 IF y>=0 AND y<=31 THEN GO SUB 4000: GO SUB 4100
3005 FOR n=0 TO -10 STEP -1: BEEP .1,n: NEXT n
3010 LET f=f-1
3015 PRINT #0;AT 1,0;"SCORE:";FN x$(sc);sc;AT 1,14;s$( TO f);" ";AT 1,21;"HSC:";FN x$(hsc);hsc
3017 IF y<0 OR y>31 THEN GO TO 3026
3020 FOR n=1 TO 4: PRINT AT x,y; INK 4; PAPER (5 AND x<10 AND x>2)+(1 AND x=1);"B": BEEP .3,-(n*3): PRINT AT x,y; INK 0; PAPER (5 AND x<10 AND x>2)+(1 AND x=1);" ": BEEP .3,-(n*3): NEXT n
3021 IF at<>57 THEN GO TO 3026
3022 IF ATTR (x,y-2)<>1 THEN PRINT AT 1,y-2; INK 1; PAPER 7;"ST ": GO TO 3026
3024 IF ATTR (x,y-1)<>1 THEN PRINT AT 1,y-1; INK 1; PAPER 7;"ST ": GO TO 3026
3025 PRINT AT 1,y; INK 1; PAPER 7;"ST "
3026 IF f>0 THEN GO TO 260
3030 FOR n=10 TO -10 STEP -1: BEEP .1,n: NEXT n
3035 DIM a$(1,32): FOR n=17 TO 20: PRINT AT n,0; PAPER 0;a$(1): NEXT n
3040 PRINT AT 21,0; INK 7;"GAME OVER": FOR n=0 TO 71: FOR m=1 TO 6: IF POINT (n,m) THEN PLOT INK 7;n*3,20+m*3
3050 NEXT m: NEXT n
3060 CLS : LET d$="SCORE:"+FN x$(sc)+STR$ sc: PRINT AT 21,0; INK 7;d$
3070 FOR n=0 TO 8*LEN d$-1: FOR m=1 TO 6: IF POINT (n,m) THEN PLOT INK 3;2*n,3*m+64
3080 NEXT m: NEXT n
3085 IF sc>hsc THEN GO TO 5000
3090 FOR n=1 TO 5: FOR m=1 TO 10 STEP 3: BEEP .1,m: NEXT m: NEXT n
3095 PRINT AT 16,8;"Highscore:";FN x$(hsc);hsc;AT 17,(32-LEN c$)/2;c$
3100 PAUSE 0: CLS : GO TO 240
4000 REM PRINTen der Strings
4010 PRINT PAPER 0; INK 2;AT 13,0;a$(5);AT 15,0; INK 7;a$(6);AT 17,0; INK 5;a$(7);AT 19,0; INK 7;a$(8)
4015 IF y<0 OR y>31 THEN GO TO 3000
4020 IF x=21 OR x<10 THEN PRINT AT x,y; INK 4; PAPER 7;"B": GO TO 4040
4030 PRINT AT x,y; PAPER 0; INK 4;"B"
4040 RETURN
4100 REM PRINTen der Strings II
4110 PRINT AT 3,0; PAPER 5;a$(1);AT 5,0;a$(2);AT 7,0;a$(3);AT 9,0;a$(4)
4115 IF y<0 OR y>31 THEN GO TO 3000
4120 IF x=21 OR x<10 THEN PRINT AT x,y; INK 4;"B": GO TO 4140
4130 PRINT AT x,y; PAPER 0; INK 4;"B"
4140 RETURN
5000 REM Highscore
5010 LET hsc=sc: PRINT AT 20,0;">>HIGHSCORE<<<": FOR n=1 TO 10: FOR m=20 TO 10 STEP -1: BEEP .005,m: NEXT m: NEXT n
5020 INPUT LINE c$: IF LEN c$>20 THEN LET c$=c$( TO 20)
5040 GO TO 3100
```

Graphic-Composer

Wer seine Sonderzeichen beim TI-99/4A per Hand entwirft, weiß ein Lied von mühseliger Kleinarbeit zu singen. Diese Arbeit übernimmt bei diesem Programm weitgehend der Computer mit Extended-BASIC

Durch Verwendung eines Menüs können sechs verschiedene „Gänge" durch Drücken der entsprechenden Tasten (1 bis 6) angewählt werden.

Genaue Anweisungen, wie man jederzeit zum Menü zurückspringen kann, werden vom Programm selbst mitgeteilt. Der gesamte Zeichensatz des Computers ist permanent am unteren Bildschirmrand sichtbar.

Die einzelnen Menüfunktionen

Grafikfeld

Auf einem Grafikfeld, das der 8×8-Matrix eines Zeichens entspricht, kann mit einem Cursor das Muster eines Sonderzeichens in groß erstellt werden. Der Computer ermittelt automatisch den CHAR-Code und definiert das Zeichen.

Scan: S. Höltgen

Vorher muß noch der Zeichentyp (ASC-Code) des betreffenden Zeichens eingegeben werden. Außerdem kann man das Muster eines vorhandenen Zeichens auf das Grafikfeld bringen. Drückt man bei der Option „Muster" die Taste ENTER ohne weitere Eingabe, so erscheint das eigene Muster des Zeichens.

Bei der Eingabe „N" erscheint kein Muster, bei „J" muß man unter „NR:" den ASC-Code des Zeichens angeben, dessen Muster erscheinen soll.

Der Cursor kann mit den vier Pfeiltasten über das Feld bewegt werden. Mit der Taste „F" kann ein Punkt gesetzt, mit SPACE gelöscht werden.

Ist das Muster fertiggestellt, wird durch ENTER die automatische Definition eingeleitet. Das zu generierende Zeichen erscheint auf dem Bildschirm, wird definiert, und der CHAR-Code wird angezeigt. Weiter geht es mit einem beliebigen Tastendruck.

Direkteingabe

Neben der Generierung per Grafikfeld können die CHAR-Codes auch direkt eingegeben werden. So können Zeichensätze aus Listings bequem in das Programm übernommen werden.

Es wird zuerst der ASC-Code des zu definierenden Zeichens unter „ASC:" und dann der CHAR-Code unter „Code:" eingegeben. Das Zeichen erscheint zur Kontrolle in derselben Zeile.

Ausdruck

Mit dieser Funktion kann man sich sämtliche CHAR-Codes des Zeichensatzes auf den Bildschirm ausdrucken lassen. Nach Eingabe der Anfangsnummer werden alle Zeichen von dieser Nummer an aufwärts ausgegeben.

Es erscheinen die Zeichen selbst, die dazugehörigen ASC-Werte und die CHAR-Codes. Per Tastendruck kann die Anzeige gescrollt werden.

Abspeichern

Sämtliche Zeichen des Computers können mit den passenden CHAR-Codes auf Kassette abgespeichert werden. Dazu gibt man unter „ASC:" den ASC-Code des Zeichens an, welches gespeichert werden soll. Das Zeichen selbst erscheint zur Kontrolle daneben.

Zum vereinfachten Speichern größerer zusammenhängender Zeichensätze wird automatisch nach der Eingabe des ASC-Codes der des nächsten Zeichens auf den Bildschirm gebracht, so daß bei längeren Serien nur noch die ENTER-Taste fortlaufend gedrückt werden muß. Gibt man den Wert „0" ein, so beginnt die Abspeicherung.

Einlesen

Die auf Kassette gespeicherten Sonderzeichen können jederzeit wieder eingelesen werden.

Für eigene Programme

Man kann die Einleseroutine des Programms „Graphic-Composer" auch in andere Programme einbauen. Dann können die Sonderzeichen, die mit „Graphic-Composer" erstellt wurden, dort eingeladen werden. Diese Routine erhält man, wenn man die Zeilen 1020 bis 1130 überträgt. Davor muß noch eine Zeile eingegeben werden, die DIM Z(100), Z$(100) enthält. Die Werte der DIM-Statements richten sich nach der Zeichenzahl.

Stephan Wiesener

```
10 !**************************
20 !*   GRAPHIC-COMPOSER     *
30 !**************************
40 !*                        *
50 !*  STEPHAN WIESENER      *
60 !*  HEINRICHSTR. 23       *
70 !*  8120 WEILHEIM         *
80 !*  TEL. 0881/62164       *
90 !*                        *
100 !**************************
110 CALL CLEAR
120 ON WARNING NEXT
130 DIM SO$(143),Z(143),Z$(143),CO(2,16)
,F(12),C$(70),M$(16),V(88),H(88)
140 !
150 !SONDERZEICHEN
160 !
170 CALL CHAR(143,"FF818181818181FF",142
,RPT$("F",16))
180 CALL CHAR(141,"FF818199998181FF")
```

```
190 !
200 !CODES FUER RUECKSETZEN
210 !
220 FOR I=96 TO 143 :: CALL CHARPAT(I,I$
):: SO$(I)=I$ :: NEXT I
230 !
240 !MUSTERSUMMANDEN
250 !
260 FOR I=5 TO 12 :: READ I1 :: F(I)=I1
:: NEXT I
270 !
280 !MUSTER-STRINGS
290 !
300 FOR I=48 TO 57 :: READ I1,I2,I3,I4 :
: C$(I)=CHR$(I1)&CHR$(I2)&CHR$(I3)&CHR$(
I4):: NEXT I
310 FOR I=65 TO 70 :: READ I1,I2,I3,I4 :
: C$(I)=CHR$(I1)&CHR$(I2)&CHR$(I3)&CHR$(
I4):: NEXT I
320 DATA 8,4,2,1,8,4,2,1
330 DATA 143,143,143,143,143,143,143,142
,143,143,142,143,143,143,142,142,143,142
,143,143
340 DATA 143,142,143,142,143,142,142,143
,143,142,142,142
350 DATA 142,143,143,143,142,143,143,142
,142,143,142,143,142,143,142,142,142,142
,143,143
360 DATA 142,142,143,142,142,142,142,143
,142,142,142,142
370 DATA 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
380 !
390 !CODESTRINGS
400 !
410 FOR I=0 TO 15 :: READ I1$ :: M$(I)=I
1$ :: NEXT I
420 !
430 !CURSORBEW.CODES
440 !
450 V(88)=1 :: V(69)=-1 :: H(83)=-1 :: H
(68)=1
460 !
470 !MENU
480 !
490 FOR I=32 TO 143 :: PRINT CHR$(I);::
NEXT I
500 DISPLAY AT(1,6):"GRAPHIC-COMPOSER":"
    STEPHAN WIESENER":"              1984"
510 CALL DELINE(4,20)
520 DISPLAY AT(5,3):"GRAPHIKFELD      <1
>":"  EINLESEN          <2>":"  ABSPEICHE
RN      <3>":"  LOESCHEN          <4>"
530 DISPLAY AT(9,3):"DIREKTEINGABE    <5
>":"  AUSDRUCK          <6>"
540 CALL KEY(0,K,S):: IF S=0 THEN 540
550 IF K>54 OR K<49 THEN 540
560 ON 55-K GOTO 1740,1910,600,700,1010,
1190
570 !
580 !LOESCHEN
590 !
600 CALL DELINE(4,15)
610 DISPLAY AT(17,1):"STANDARDZEICHENSAT
Z?(J/N) J"
620 ACCEPT AT(17,27)SIZE(-1)VALIDATE("JN
")BEEP:L$
630 IF L$="J" THEN 640 ELSE 660
640 CALL CHARSET
650 FOR I=96 TO 143 :: CALL CHAR(I,SO$(I
)):: NEXT I
660 GOTO 510
670 !
680 !ABSPEICHERN
690 !
700 CALL DELINE(4,20)
710 DISPLAY AT(19,1):"  SPEICHERN: 0 EIN
GEBEN!"
720 Y=7 :: ANZ=1
730 DISPLAY AT(5,3):"ASC:   TYP:"
740 ACCEPT AT(Y,3)SIZE(-3)BEEP:Z(ANZ)
750 IF Z(ANZ)=0 THEN ANZ=ANZ-1 :: GOTO 8
20
760 IF Z(ANZ)<32 OR Z(ANZ)>143 THEN 740
770 DISPLAY AT(Y,11)SIZE(2):CHR$(Z(ANZ))
780 Y=Y+1 :: IF Y>17 THEN Y=7
790 DISPLAY AT(Y,2)SIZE(4):Z(ANZ)+1
800 ANZ=ANZ+1 :: IF ANZ>150 THEN ANZ=1
810 GOTO 740
820 CALL SOUND(700,120,10,240,0)
```

Scan: S. Höltgen


```
830 FOR I=1 TO ANZ
840 CALL CHARPAT(Z(I),Z$(I))
850 NEXT I
860 CALL CLEAR
870 OPEN #1:"CS1",SEQUENTIAL,INTERNAL,OU
TPUT,FIXED
880 PRINT #1:ANZ
890 FOR I=1 TO ANZ STEP 7
900 PRINT #1:Z(I),Z(I+1),Z(I+2),Z(I+3),Z
(I+4),Z(I+5),Z(I+6)
910 NEXT I
920 FOR I=1 TO ANZ STEP 3
930 PRINT #1:Z$(I),Z$(I+1),Z$(I+2)
940 NEXT I
950 CLOSE #1
960 CALL CLEAR
970 GOTO 490
980 !
990 !EINLESEN
1000 !
1010 CALL CLEAR
1020 OPEN #1:"CS1",SEQUENTIAL,INTERNAL,I
NPUT ,FIXED
1030 INPUT #1:ANZ
1040 FOR I=1 TO ANZ STEP 7
1050 INPUT #1:Z(I),Z(I+1),Z(I+2),Z(I+3),
Z(I+4),Z(I+5),Z(I+6)
1060 NEXT I
1070 FOR I=1 TO ANZ STEP 3
1080 INPUT #1:Z$(I),Z$(I+1),Z$(I+2)
1090 NEXT I
1100 CLOSE #1
1110 FOR I=1 TO ANZ
1120 CALL CHAR(Z(I),Z$(I))
1130 NEXT I
1140 CALL CLEAR
1150 GOTO 490
1160 !
1170 !GRAPHIKFELD
1180 !
1190 CALL DELINE(4,20)
1200 DISPLAY AT(19,1):"  MENU: 0 EINGEBE
N!"
1210 FOR I=8 TO 15 :: DISPLAY AT(I,3)SIZ
E(8):RPT$(CHR$(143),8):: NEXT I
1220 DISPLAY AT(7,3):"84218421"
1230 DISPLAY AT(17,3):"TYP:"
1240 DISPLAY AT(8,13):"MUSTER:  NR.:"
1250 DISPLAY AT(10,13):"CODE:"
1260 ACCEPT AT(17,7)SIZE(3)VALIDATE(DIGI
T)BEEP:TYP :: IF TYP=0 THEN 510 ELSE IF
TYP<32 OR TYP>143 THEN 1260
1270 DISPLAY AT(17,11)SIZE(2):CHR$(TYP)
1280 ACCEPT AT(8,20)SIZE(1)VALIDATE("JN"
)BEEP:MUSTER$
1290 IF MUSTER$="J" THEN ACCEPT AT(8,26)
SIZE(3):NR ELSE IF MUSTER$="" THEN NR=TY
P ELSE 1400 :: IF NR<32 OR NR>143 THEN 1
290
1300 CALL CHARPAT(NR,CODE$)
1310 SEG=1
1320 FOR I=8 TO 15
1330 DISPLAY AT(I,3)SIZE(4):C$(ASC(SEG$(
CODE$,SEG,1)))
1340 DISPLAY AT(I,7)SIZE(4):C$(ASC(SEG$(
CODE$,SEG+1,1)))
1350 SEG=SEG+2
1360 NEXT I
1370 !
1380 !CURSOR
1390 !
1400 Y=8 :: X=5
1410 FOR T=1 TO 100 :: NEXT T
1420 CALL GCHAR(Y,X,G)
1430 CALL HCHAR(Y,X,141)
1440 CALL KEY(0,K,S):: IF S=0 THEN 1450
ELSE IF K=13 THEN 1550 ELSE 1460
1450 CALL HCHAR(Y,X,G):: FOR T=1 TO 10 :
: NEXT T :: CALL HCHAR(Y,X,141):: FOR T=
1 TO 10 :: NEXT T :: GOTO 1440
1460 IF K=32 THEN CALL HCHAR(Y,X,143)::
GOTO 1420 ELSE IF K=70 THEN CALL HCHAR(Y
,X,142):: GOTO 1420
1470 YY=Y+V(K):: XX=X+H(K)
1480 IF XX<5 OR XX>12 OR YY<8 OR YY>15 T
HEN CALL HCHAR(Y,X,G):: GOTO 1420
1490 CALL HCHAR(Y,X,G)
1500 Y=YY :: X=XX
1510 GOTO 1420
1520 !
1530 !DEFINITION
1540 !
1550 CALL HCHAR(Y,X,G)
1560 CALL SOUND(-400,120,10,240,15)
1570 FOR X=5 TO 12
1580 FOR Y=8 TO 15
1590 CALL GCHAR(Y,X,G)
1600 IF G=142 THEN 1610 ELSE 1620
1610 IF X>8 THEN CO(2,Y)=CO(2,Y)+F(X)ELS
E CO(1,Y)=CO(1,Y)+F(X)
1620 NEXT Y
1630 NEXT X
1640 FOR Y=8 TO 15
1650 PAT$=PAT$&M$(CO(1,Y))&M$(CO(2,Y))
1660 NEXT Y
1670 CALL CHAR(TYP,PAT$)
1680 FOR Y=8 TO 15 :: CO(1,Y)=0 :: CO(2,
Y)=0 :: NEXT Y
1690 DISPLAY AT(11,13)SIZE(16):PAT$ :: P
AT$=""
1700 CALL KEY(0,K,S):: IF S=0 THEN 1700
ELSE DISPLAY AT(11,13)SIZE(16):"
    " :: GOTO 1210
1710 !
1720 !AUSDRUCK
1730 !
1740 CALL DELINE(4,20)
1750 DISPLAY AT(19,1):"  MENU: REDO DRUE
CKEN!"
1760 DISPLAY AT(5,3):"ANFANGSNR.:32"
1770 ACCEPT AT(5,14)SIZE(-3)BEEP VALIDAT
E(DIGIT):ANF
1780 IF ANF<32 OR ANF>143 THEN 1770
1790 FOR I=6 TO 17
1800 CALL CHARPAT(ANF,CODE$)
1810 DISPLAY AT(I,2):CHR$(ANF);ANF;CODE$
1820 ANF=ANF+1
1830 IF ANF>143 THEN ANF=32
1840 NEXT I
1850 CALL KEY(0,K,S)
1860 IF S<>0 THEN 1870 ELSE 1850
1870 IF K=6 THEN 510 ELSE 1790
1880 !
1890 !DIREKTEINGABE
1900 !
1910 CALL DELINE(4,20)
1920 DISPLAY AT(19,1):"MENU: 0 EINGEBEN!
"
1930 DISPLAY AT(5,1):"ASC:      CODE:"
1940 I=7
1950 ACCEPT AT(I,1)SIZE(3)VALIDATE(DIGIT
):NR :: IF NR=0 THEN 510 ELSE IF NR<32 O
R NR>143 THEN 1950
1960 DISPLAY AT(I,5)SIZE(1):CHR$(NR)
1970 ACCEPT AT(I,10)SIZE(16)VALIDATE("12
34567890ABCDEF"):CODE$
1980 CALL CHAR(NR,CODE$)
1990 I=I+1 :: IF I>17 THEN I=7
2000 GOTO 1950
2010 SUB DELINE(A,B)
2020 FOR I=A TO B :: CALL HCHAR(I,1,32,3
2):: NEXT I :: SUBEND
```


Software und Hardware von

Commodore C 64 **„Dynamics"** TI 99/4a ATARI

Dynamics-Versand W. Bauernfeind

Hamburger Str. 78 2359 Henstedt-Ulzburg Telefon 0 41 93/77 05

Scan: S. Höltgen

Für jeden etwas...

Wir haben vier Spiele ausgesucht und für Sie getestet

Donkey Kong jr.

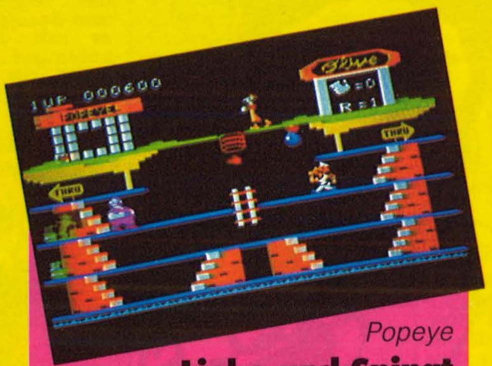
Der Kleine vom großen Affen

Computer-Spieler wissen es längst: Donkey Kong war ein Riesenerfolg. Und die Erfolgssträhne hält noch immer an. Entwickelt wurde das Spiel, das von Atari vermarktet wird, in Japan. Der Spiele-Gigant Nintendo steckt wie bei anderen erfolgreichen Programmen auch hinter dieser Entwicklung. Nochmals zu Donkey Kong – dem Senior. Bekanntlich bekam Mario, der unglückliche Liebhaber, dem sein Mädchen vom Gorilla geraubt worden ist, nach waghalsigen Befreiungsmanövern seine Herzensallerliebste wieder zurück. Aber das war dem so hart geforderten Mario nicht genug. Er nahm Donkey Kong gefangen. Nun muß der Kleine vom großen Affen seinen Papi retten. Donkey Kong junior also auf dem Befreiungstrip für den Senior. Der Junior beginnt das Spiel mit den drei Leben. Er bekommt für ein Extraleben 10000 Punkte. Eine gewaltige Summe. Allerdings gilt diese Prämie für das Punktekonto nur solange, wie Donkey Kong junior am Leben bleibt. Das Rettungsmanöver spielt sich in vier Szenen ab. Ein Theaterregisseur scheint hier Regie geführt zu haben. Es gibt die Lianen-Szene, die Ketten-Szene, die Sprungbrett-Szene und die Schlupfwinkel-Szene. Um ein Beispiel zu schildern. In der Lianenszene muß sich Donkey Kong Junior den Weg zu den Schlüsseln durch Klettern, Springen und Beißen erkämpfen. Auf seinem Weg kann er Früchte pflükken, um Punkte zu machen, die Früchte auf die „Beißer" fallen lassen, um noch mehr Punkte auf seinem Konto zu vereinen. Einige „Beißer" rutschen die Lianen hinunter und fallen herab, während andere sich umdrehen und wieder hinaufklettern. Ein richtiges Urwaldspektakel.

Schwierigkeitsgrad: Steigt von Szene zu Szene.

Unser Testurteil: „They never come back" ist ein geflügeltes Wort bei Boxkämpfen geworden. Wer einmal groß war, wird nie mehr so groß den Boxring verlassen. Läßt sich das auch auf Spiele anwenden? Kann man also in einem Quasi-Aufguß – in diesem Fall wohl eine Fortsetzung – noch soviel Spannung packen, daß der Computer-Spieler von der Faszination eingefangen wird? Nach den Erfahrungen mit Donkey Kong möchten wir behaupten: „They come back". Donkey Kong junior ist ein faszinierendes Spiel.

Hersteller: Nintendo für Atari
Geeignet: für Atari-Computer
Preis: 99 Mark



Popeye

Liebe und Spinat

Da haben sich die Software-Spezialisten wieder einmal einen besonderen Gag einfallen lassen: Die Jagd nach Liebesherzen. Das ist kein Spaß, sondern Bildschirmwirklichkeit. Stellen Sie sich vor, Sie verkörpern im Spiel Popeye und sind bis über beide Ohren in Olive verliebt. Die signalisiert ihre Zuneigung mit Noten und Herzen. Ihre Aufgabe ist es, alle Herzen, Noten und Hilfeschreie aufzufangen, bevor sie ins Wasser fallen und versinken. Aber das ist gar keine leichte Aufgabe. Gibt es doch noch ein paar finstere Gesellen – Brutus und Sea Hag –, die anscheinend auf Popeyes Liebesleben eifersüchtig sind. Sie stören ständig und trachten Popeye sogar nach dem Leben. Besonders gemein benimmt sich dabei Sea. Der wirft mit Flaschen. Einzige Rettung für Popeye: Er muß die Wurfgeschosse eigenhändig zerschmettern. Brutus wiederum ist ein Faustkämpfer. Und wehe, wenn sie Popeye trifft. Der verliebte Wicht kann sich allerdings ähnliche Kräfte wie Obelix und Asterix mit dem Zaubertrank zulegen. Er muß nur eine Portion Spinat essen, die gelegentlich im Spiel auftaucht. Dann ist er für einige Zeit nicht verwundbar. Übrigens müssen in jeder Runde eines Spieles 24 Herzen von Popeye gefangen werden.

Schwierigkeitsgrad: Die Anforderungen an den Spieler steigen mit dem Fortgang des Spieles.

Spieler: 1 und 2

Unser Testurteil: Programme, die ein Gerüst im Spiel haben, sind in den letzten eineinhalb Jahren häufig auf den Markt gekommen. Meist vollzieht sich dann auf dem Gerüst die wilde Jagd nach einem Gorilla, der ein Mädchen verschleppt und seine Verfolger mit allen möglichen Wurfgeschossen abzuschütteln versucht. Bei Popeye hat sich der Programmierer eine sehr liebenswerte Version einfallen lassen. Wo gibt es denn schon Herzen in einem Computerspiel? Damit ist das Spiel gelungen, das vom Kleinkind bis zur Großmutter ankommen dürfte. Den Gag mit dem Spinat kennen passionierte Computerspieler bereits von PacMan, der al-

Scan: S. Höltgen

Spiele Test

lerdings seine übermenschlichen Kräfte von Keksen bezieht. Als Plus dieses Spiels ist das langsame, fast übergangslose Steigern des Schwierigkeitsgrades zu bezeichnen. Leider wurde uns das Spiel nur mit einer englischsprachigen Anleitung geliefert.
Hersteller/Vertrieb: Texas Instruments, Wolfgang Riegert, 7324 Rechberghausen.
Geeignet: für Texas Instruments TI-99/4A.
Preis: 89 Mark.

Star Ranger

Raumschiffpilot

Wieder ein Weltraumspiel: Star Ranger versetzt Sie in die Position eines Raumfahrschiffes, dessen Aufgabe es ist, nach einem Aufklärungsflug wieder sicher auf der Mondbasis zu landen. Vor Ihrem Start wird Ihnen schon gesagt, daß Sie mit einer schwierigen Mission beauftragt wurden. Und so kommt es dann auch. Kaum haben Sie die Triebwerke außer Sichtweite der Basis getragen, erscheinen schon die ersten feindlichen Raumschiffe auf Ihrem Radarschirm. Zu Ihrem Erstaunen handelt es sich gleich um eine Vielzahl verschiedener Flugkörper. Die meisten haben Sie noch nie gesehen. Sie müssen also ohne Training die Initiative ergreifen. Hier gilt es, sich wie ein ausgefuchster Pilot zu verhalten. Nur gekonnte Flugmanöver, die mit viel Fingerspitzengefühl am Joystick ausgeführt werden, verhindern, daß man ständig mit den feindlichen Flugkörpern kollidiert. Außerdem muß ein guter Schütze am Steuerknüppel sitzen, um in diesem Inferno des Weltraumkrieges bestehen zu können. Selbstverständlich wird die Leistung gemessen. Der Weltraumpilot muß also zeigen, was er kann.

Schwierigkeitsgrad: Das Spiel bietet sechs Schwierigkeitsgrade an, wobei 1 als leichtester und 6 als schwierigster gelten.
Spieler: 1
Unser Testurteil: Da scheint ein ausgefuchster Programmierer am Werk gewesen zu sein. Die sechs Schwierigkeitsgrade lassen kaum ein Aufatmen zu. Nach unseren Tests meinen wir, daß der Schwierigkeitsgrad 6 auch Profis unter den Computer-Spielern noch herausfordern kann. Gut gelungen scheint uns die Bewertungstabelle zu sein. Hier hat man einen Überblick der geforderten und der schließlich gebrachten Leistung. Die Spielanweisung – die uns nur in englisch vorlag – bringt auch einige Hinweise, wie man sich am Weltraumhimmel des Bildschirms durchsetzt. Commodore-64-Fans werden ihre Freude an diesem Spiel haben.
Hersteller: Commodore
Geeignet: für Commodore C 64
Preis: 59 Mark

Bigfoot

Vorsicht: das Bergmonster kommt

Stellen Sie sich vor, Sie möchten im Himalaja einen Achttausender bezwingen. Lange haben Sie in den Alpen Europas dafür trainiert. Ihre Kondition reicht für das schwierige Unternehmen. Außerdem haben Sie sich über Jahre hinweg mit den Gefahren der Berge vertraut gemacht. Eigentlich könnte Ihnen nichts mehr passieren. Am Anfang geht auch alles gut. Keiner der mitsteigenden Sherpas verletzt sich oder stürzt ab. Sie erreichen heil das Basislager und schicken die Träger wieder zurück. Jetzt wollen Sie den Achttausender im Alleingang bezwingen. Die Einsamkeit der Berge fängt Sie ein. Doch plötzlich hören Sie ein gewaltiges Grollen. Sie glauben zuerst an ein Gewitter. Doch dann sehen Sie oben auf einem Felsvorsprung eine riesige weiße Gestalt – den sagenumwobenen Schneemenschen Yeti. Der scheint Ihnen den Aufstieg übelzunehmen. Als Sie nämlich nicht auf sein Grollen reagieren, fängt er an, Felsbrocken nach unten zu werfen. Seine Absicht ist schnell durchschaut. Er will

Sie mit einem Treffer die Wand hinunterstürzen sehen. Am Anfang gelingt es Ihnen ziemlich leicht, den herabrollenden Gesteinsbrokken auszuweichen. Sie könnten jetzt selbstverständlich umkehren. Aber das liegt nicht in Ihrer Absicht. Je weiter Sie nach oben kommen, um so schwieriger wird es, den heimtückischen Wurfgeschossen auszuweichen. Schaffen Sie den Gipfel?
Schwierigkeitsgrad: Steigt mit dem Fortgang des Spiels. Je weiter der Spieler nach oben kommt, um so schwieriger werden Ausweichmanöver.
Spieler: 1
Unser Testurteil: Wir kennen bisher ein Spiel dieser Art nur aus dem Videobereich. Nach den vielen Labyrinth- und Weltraumspielen ist dieses eine willkommene Abwechslung in der sich anbahnenden Monotonie bei den Computerspielen. Wir haben schon lange nicht mehr so lange vor dem Bildschirm gesessen. Der Programmierer dieses Spiels macht es dem Spieler wirklich nicht leicht. Wer gibt schon gern kurz vor dem Gipfel auf. Wir standen oft vor dem Ziel und mußten uns dann vom „Schneemenschen" geschlagen geben. Das erhöht die Motivation. So kommen schnell viele Stunden vor dem Bildschirm zusammen.
Hersteller: Milton Bradley Company für Texas Instruments. Vertrieb: Radix Bürotechnik, Bornstraße 4, 2000 Hamburg 13
Geeignet: für TI-99/4A
Preis: 99 Mark

Scan: S. Höltgen


```
 9690
9600 IF y=14 AND x=11 THEN GO TO
 9620
9610 GO TO 9580
9620 CLS : PRINT AT 10,2;"Besitz
t dieser Drucker eine";AT 12,6;"
Hardcopy funktion ?";AT 14,11; I
NVERSE 1;"J"; INVERSE 0;"a / ";
INVERSE 1;"N"; INVERSE 0;"ein"
9630 LET y=14: LET x=16
9640 GO SUB 8000
9650 IF y=14 AND x=16 THEN GO TO
 9690
9660 IF y=14 AND x=11 THEN GO TO
 9680
9670 GO TO 9640
9680 INPUT "Einsprungadresse fue
r Hardcopy: ";p1
9690 DIM d$(3,20): REM <<Grosse
Uhranzeige definieren>>
9700 LET d$(1)="▛▌▟ ▀▜▀▜▐▐▛▀▛▀▀▜
▛▌▛▌"
9710 LET d$(2)="▐▐ ▐ ▛▀▀▜▀▜▀▜▛▌ ▐
▛▌▀▜"
9720 LET d$(3)="▄▄ ▄▄ ▄▄ ▄▄  ▄ ▄▄ ▄▄  ▄
▄▄ ▄▄"
9725 REM <<Selbstdefinierte Zeic
hen Codes>>
9730 DATA 0,15,8,9,8,40,24,9,0,2
48,0,16,160,64,160,16,0,16,48,16
,17,17,58,2,64,64,162,148,8,20,3
4,0,0,112,8,48,64,120,0,0,0,0,0,
68,40,16,40,68,0,68,40,16,16,0,0
,0,0,0,63,72,8,8,16,0,0,8,240,64
,64,64,32,0,0,0,16,16,124,17,18,
0,0,16,32,64,158,0,0,0
9740 DATA 3,4,248,144,159,131,13
1,135,192,32,31,9,249,225,249,25
3,135,143,143,159,159,191,143,13
1,253,249,249,241,241,225,225,19
3,128,128,128,128,128,128,128,25
5,129,1,1,1,1,1,1,255,255,128,13
2,148,148,149,149,159,255,1,17,1
7,81,81,81,249,128,140,146,161,1
61,146,140,128,1,125,1,125,1,125
,1,1,128,144,169,165,133,130,128
,255,1,193,37,21,9,1,1,255,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,60,0,25
5,128,255,128,255,128,188,66,189
,67,189,67,129,1,1,128,188,128,1
88,128,255,0,0,1,1,1,1,1,255,0,0
9750 DATA 127,129,128,129,129,12
9,129,199,252,2,2,2,2,2,2,134,12
9,128,128,128,128,129,127,0,2,2,
2,2,2,2,252,0,15,31,17,17,17,17,
31,0,248,248,24,24,24,24,240,0,3
1,63,127,255,128,159,160,160,254
,253,251,247,15,207,47,47,160,17
5,160,159,128,255,0,255,47,175,4
7,206,12,248,0,255,255,170,255,2
13,255,202,255,0,255,171,255,85,
255,169,255,0,63,64,95,80,95,64,
64,91,254,1,253,5,253,1,1,109,91
,64,91,91,64,91,91,64,109,1,109,
109,1,109,109,1,91,91,64,91,91,9
1,64,63,109,109,1,109,109,109,1,
254
9790 CLS : PRINT AT 10,10;"Bitte
 Warten."
9800 RESTORE 9730: FOR i=65280 T
O 65367: READ a: POKE i,a: NEXT
i
9810 RESTORE 9740: FOR i=65024 T
O 65167: READ a: POKE i,a: NEXT
i
9820 RESTORE 9750: FOR i=64768 T
O 64911: READ a: POKE i,a: NEXT
i
9830 GO TO 0
```

Defender

Abfangjäger versuchen mit Laser-Kanonen, Angreifer aus dem Weltraum abzuwehren (TI-99/4A mit Extended-BASIC und Joystick)

Es stehen zunächst drei Abfangjäger zur Verfügung; bei einer bestimmten Punktzahl erhält man als Bonus einen zusätzlichen. Sie stehen auf einer Startrampe und können mittels Joystick ihre Position verändern; mit der Feuertaste kann geschossen werden. Die Angreifer setzen sich aus vier Arten von Ufos zusammen. Beim Eintippen ist darauf zu achten, daß Klein- und Großschreibweise sowie die Umlaute genau wie im Listing eingegeben werden. Wird das Programm ohne Erweiterungssystem benutzt, müssen die Zeilen 420 und 440 gelöscht werden. *Hans-Peter Kreutz*

```
10 RANDOMIZE
20 CALL CLEAR
30 REM ZEICHENDEFINITION----------------
----------------------------------------
40 S3$="0042001208029201 0D52000A12009204
00200410218029C09082056000490064"
50 S4$="00000001030A376FF07F3F13214347E0
00000080C050ECF60FFEFCC884C2E207"
60 S5$="00000036728FFFFFFFF3F6F172340E0
00000083C62DF8FFFFFFBFDE6D388040E"
70 S6$="0000000000007FFFFF7F000000000000
00000183870F0F6FEFE6F07038180000"
80 S7$="173163D78F183FFFFE3E1E0F060C183C
E88CC6EBF118FCFF7F7C78F060301 83C"
90 S2$="0000FF00000000000000000000FF0000
0000E0000000000000000000000E00000"
100 S1$="040281473F95D5F3D5953F4F8103070
0204080C0EFF85C3E7E5CF8EF0080C000"
110 S8$="03000000FD070F383F0F07FD0302070
7F84141FAFCFDFFFFDFDFDDFCEA091D1C"
120 S9$="1F011D090B0F1CFFFF1C070F193367C
0F880B890D0F038FFFF38E0F098CCE6D3"
130 S11$="6882808083068F4D4D2B0513080200
0000008020C0E8B470D2B060C001018136"
140 S13$="FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF"
150 S12$="071F3C7C48EBABF3FDC4E4446C361B
07E0B88C86CEE3EBFF7323F7FA3E3CF8E0"
160 S14$="C0808CFC9CBCF8B092FFFF8890A0C0
```

Scan: S. Höltgen


```
800301313F393D1F0D49FFFF1109050301"
170 S10$="000101"
180 CALL CHAR(132,S3$):: CALL CHAR(104,S
8$):: CALL CHAR(100,S4$):: CALL CHAR(108
,S5$):: CALL CHAR(112,S6$):: CALL CHAR(1
16,S7$):: CALL CHAR(124,S13$)
190 CALL CHAR(128,S10$):: CALL CHAR(40,S
11$):: CALL CHAR(36,S12$)
200 CALL CHAR(120,S9$):: CALL CHAR(64,S1
4$)
210 CALL CHAR(96,S2$)
220 CALL CHAR(92,S1$)
230 CALL CHAR(125,"FF"):: CALL CHAR(126,
"00000000000000FF"):: CALL CHAR(127,"3C4
299A1A199423C")
240 CALL SPRITE(#4,100,8,190,25,10,-25)
250 CALL MAGNIFY(3)
260 REM SPIELEINWEISUNG-----------------
----------------------------------------
270 E=30 :: F=-20 :: CALL CLEAR :: CALL
SCREEN(2):: CALL SPRITE(#3,100,4,75,50,3
0,35):: FOR N=2 TO 12 :: CALL COLOR(N,4,
1):: NEXT N
280 CALL HCHAR(1,2,125,30):: CALL HCHAR(
24,2,126,30)
290 DISPLAY AT(2,7):"  ~ DEFENDER ~"
300 DISPLAY AT(4,3):CHR$(127)&" KREUTZ,H
.-PETER,ULM/DO."
310 DISPLAY AT(6,3):"POINTS:"
320 DISPLAY AT(8,3):"df" :: DISPLAY AT(9
,3):"eg            10 POINTS"
330 DISPLAY AT(11,3):"tv" :: DISPLAY AT(
12,3):"uw              50 POINTS"
340 DISPLAY AT(14,3):"hj" :: DISPLAY AT(
15,3):"ik            100 POINTS"
350 DISPLAY AT(17,3):"xz" :: DISPLAY AT(
18,3):"y{            100 POINTS"
360 DISPLAY AT(20,3):"ln" :: DISPLAY AT(
21,3):"mo            150 POINTS"
370 DISPLAY AT(23,15):" PRESS KEY   >>" :
: CALL KEY(0,KEY,STA):: IF STA=0 THEN 37
0
380 CALL CHAR(92,S1$)
390 REM SCENENAUFBAU/VORSPANN-----------
----------------------------------------
400 CALL SCREEN(2):: CALL CLEAR
410 DISPLAY AT(12,7):"~ ONE MOMENT ~" ::
 FOR I=1 TO 450 :: NEXT I :: CALL CLEAR
:: FOR N=3 TO 12
420 CALL COLOR(N,14,1):: NEXT N :: FOR N
=2 TO 26 :: CALL SPRITE(#N,128,1,96,120,
RND*47-15,RND*39-15):: NEXT N
430 REM START DER UFO'S VOM FREMDEN PLAN
ETEN------------------------------------
440 CALL INIT :: CALL LOAD(-31878,1):: F
OR N=2 TO 26 :: CALL LOCATE(#N,96,120)::
 CALL COLOR(#N,INT(N/2)+2):: NEXT N
450 CALL SPRITE(#15,100,5,96,120,30,-33)
:: CALL SPRITE(#17,104,4,96,120,37,25)::
 CALL SPRITE(#18,116,14,96,120,-30,30)
460 CALL SPRITE(#20,92,8,96,120,-23,-30)
470 CALL LOAD(-31878,26):: CALL SPRITE(#
22,40,6,96,120,0,0):: FOR I=0 TO 30 :: C
ALL SOUND(50,-6,I):: NEXT I
480 CALL SCREEN(2):: CALL CLEAR
490 FOR I=1 TO 30 :: DISPLAY AT(RND*24+1
,RND*28+1)SIZE(1):"." :: NEXT I
500 HIGH=500 :: PA=104 :: PA1=96 :: CO1=
11 :: CO2=15 :: SC1=100 :: SC2=10
510 CALL PATTERN(#4,100):: CALL COLOR(#4
,6)
520 REM HAUPTPROGRAMMTEIL***************
****************************************
530 CALL DELSPRITE(#17):: CALL DELSPRITE
(#18):: CALL DELSPRITE(#20):: CALL SPRIT
E(#22,40,6,96,120,-4,6)
540 DISPLAY AT(1,1):"\^\^\^\^     POINTS:
" :: DISPLAY AT(2,1):"]_]_]_]_" :: DISPL
AY AT(1,22):SCORE :: FOR K=0 TO 3
550 DISPLAY AT(22,11):"@B" :: DISPLAY AT
(23,11):"AC"
560 DISPLAY AT(1,2*K+1)SIZE(2):"  " :: D
ISPLAY AT(2,2*K+1)SIZE(2):"  "
570 CALL COLOR(13,5,1)
580 CALL SPRITE(#15,92,6,170,16,0,-4)::
CALL SPRITE(#16,100,14,168,87,0,-4)
590 ZAE=0 :: E=-30+INT(RND*60):: F=-20+I
NT(RND*40):: CALL SPRITE(#1,92,3,156,97,
0,0)
600 CALL SPRITE(#11,37,6,80,170,2,4):: C
ALL SPRITE(#14,40,7,40,60,0,3)
610 CALL HCHAR(24,1,124,32)
620 CALL SPRITE(#12,40,14,50,170,2,4)::
CALL SPRITE(#13,37,8,140,100,0,2)
630 CALL SPRITE(#8,PA,CO1,1,230)
640 CALL JOYST(2,XPOS,YPOS)
650 CALL MOTION(#1,-YPOS*5,XPOS*5)
660 REM ANGRIFF DER UFO'S---------------
----------------------------------------
670 CALL MOTION(#8,E,F)
680 REM FEUERBEFEHL---------------------
----------------------------------------
690 CALL KEY(2,TAS,STATUS):: IF TAS=18 T
HEN 700 ELSE 760
700 REM FEUER---------------------------
----------------------------------------
710 CALL SOUND(150,-7,0)
720 CALL POSITION(#1,X,Y):: CALL SPRITE(
#2,96,16,X,Y,0,127)
730 CALL POSITION(#2,X1,X2)
740 IF X2<200 THEN 750 ELSE CALL DELSPRI
TE(#2):: GOTO 760
750 CALL COINC(#2,#8,25,CO):: CALL COINC
(#2,#4,20,C24):: IF CO=0 AND C24=0 THEN
730 ELSE 990
760 CALL JOYST(2,XPOS,YPOS):: CALL MOTIO
N(#1,-YPOS*5,XPOS*5)
770 REM ANGRIFF RAUMF[HRE---------------
----------------------------------------
780 CALL MOTION(#4,-30+INT(RND*60),-20+I
NT(RND*40))
790 REM COLLISION-----------------------
----------------------------------------
800 CALL COINC(#1,#8,20,C18):: IF C18=-1
 THEN 950
810 REM COLLISION-----------------------
----------------------------------------
820 CALL COINC(#1,#4,20,C14):: IF C14=-1
 THEN 950
830 E=0 :: F=0
840 REM ANGRIFF UFO---------------------
----------------------------------------
850 IF ZAE=2 THEN E=-50+INT(RND*100):: F
=-40+INT(RND*80):: ZAE=0 :: GOTO 640
860 CALL JOYST(2,XPOS,YPOS):: CALL MOTIO
N(#1,-YPOS*5,XPOS*5)
870 IF INT(RND*2)=1 THEN 890 ELSE 640
880 REM GEGNERISCHES FEUER--------------
----------------------------------------
890 CALL POSITION(#8,X8,Y8):: CALL SPRIT
E(#9,PA1,CO2,X8,Y8,0,-127)
900 CALL SOUND(150,-5,0)
910 CALL POSITION(#9,X9,Y9):: IF Y9>55 T
HEN 920 ELSE CALL DELSPRITE(#9):: ZAE=ZA
E+1 :: GOTO 640
920 CALL COINC(#1,#9,25,CO):: IF CO=-1 T
HEN 950
930 CALL JOYST(2,XPOS,YPOS):: CALL MOTIO
```

Scan: S. Höltgen


```
N(#1,-YPOS*5,XPOS*5):: GOTO 910
940 REM EIGENE RAUMF[HRE ZERST\RT-------
----------------------------------------
950 CALL DELSPRITE(#9):: FOR X=13 TO 16
:: CALL SCREEN(X):: CALL PATTERN(#1,132)
:: NEXT X :: CALL SCREEN(2):: IF C18=0 T
HEN 960 ELSE CALL PATTERN(#8,132):: CALL
 SOUND(50,-6,X)
960 CALL MOTION(#1,0,0):: CALL MOTION(#8
,0,0):: FOR I=0 TO 30 :: CALL SOUND(50,-
6,I):: NEXT I
970 NEXT K :: GOTO 1100
980 REM GEGNERISCHES RAUMSCHIFF ZERST\RT
----------------------------------------
990 CALL DELSPRITE(#2):: IF C24=-1 THEN
1040 ELSE CALL PATTERN(#8,132)
1000 CALL MOTION(#1,0,0):: CALL MOTION(#
8,0,0):: FOR I=0 TO 30 :: CALL SOUND(100
,-7,I):: NEXT I :: SCORE=SCORE+SC1
1010 DISPLAY AT(1,22):SCORE :: IF SCORE<
200 THEN PA=104 ELSE PA=108 :: PA1=112 :
: CO1=10 :: CO2=7 :: SC1=150
1020 GOTO 590
1030 REM UFO ZERST\RT-------------------
----------------------------------------
1040 CALL PATTERN(#4,132):: FOR I=0 TO 3
0 :: CALL SOUND(50,-3,I):: NEXT I :: CAL
L SCREEN(2):: SCORE=SCORE+SC2 :: CALL PA
TTERN(#4,100):: DISPLAY AT(1,22):SCORE
1050 IF SCORE>=HIGH THEN GOSUB 1170
1060 IF SCORE>600 AND SC2=50 THEN CALL P
ATTERN(#4,120):: SC2=100 :: CALL COLOR(#
4,6):: GOTO 640
1070 GOTO 640
1080 GOTO 640
1090 REM SPIELENDE**********************
****************************************
1100 CALL CLEAR :: CALL DELSPRITE(ALL)::
 CALL SCREEN(2)
1110 DISPLAY AT(12,8):"NEW ? Y  OR  N
  N" :: ACCEPT AT(12,27)SIZE(-1)VALIDATE
("YN")BEEP:ST$
1120 HSC=MAX(HSC,SCORE):: DISPLAY AT(15,
8):"HIGTHPOINTS  ";HSC :: SCORE=0
1130 FOR I=1 TO 1000 :: NEXT I :: IF ST$
="Y" THEN 1140 :: IF ST$="N" THEN 1150 :
: CALL CLEAR
1140 GOTO 410
1150 CALL CLEAR :: END
1160 REM BONUS--------------------------
----------------------------------------
1170 DISPLAY AT(1,2*K+1)SIZE(2):"\^" ::
DISPLAY AT(2,2*K+1)SIZE(2):"]_" :: K=K-1
 :: HIGH=HIGH+2000 :: RETURN
```

Zensur

Alle bei der Auswertung einer Klassenarbeit wünschenswerten Hilfestellungen gibt der ZX 81 mit mindestens 16K und Drucker. Es lassen sich sogar mehrere Vorschläge des Rechners mit jeweils anderen Notengrenzen durchspielen

Pro Klassenarbeit sind höchstens neun Aufgaben mit jeweils maximal neun Punkten möglich. Es gibt keine halben Punkte. Es werden keine geteilten Noten (zum Beispiel 3/4), keine 1+, keine 6+ und keine 6− vergeben. Tendenznoten (zum Beispiel 3+ oder 4−) werden nur auf den Punktbereichsgrenzen erteilt.

Eingaben

Die meisten Eingaben werden durch ein Blinksignal angefordert. Sie werden entweder vom Anwender durch NEW LINE abgeschlossen, oder der Rechner bricht sie beim Erreichen einer vorgegebenen maximalen Zeichenanzahl selbsttätig ab. Jede Eingabe ist an einen Plausibilitätstest gekoppelt, der unsinnige oder nicht erlaubte Eingaben zurückweist und sofort neu anfordert. Damit wird verhindert, daß das Programm durch einen Eingabefehler abbricht.

Das Menü

Unmittelbar nach dem Starten des Programmes erscheint auf dem Bildschirm das Menü. Durch Drücken irgendeiner Taste wird das Angebot ausgewählt, dessen Blinksignal gerade tätig ist. In den meisten Fällen wird dies das Angebot „Arbeit“ sein. Die drei anderen Angebote kommen naturgemäß wohl nur am Ende eines Schuljahres oder bei der ersten Benutzung des Programmes in Frage.

```
ARBEIT               ▒


EINSTUFUNG           ▒
AUSSTUFUNG           ▒
NEUE KLASSE          ▒
```

Das Programm soll an folgendem Beispiel vorgestellt werden: Die Klasse hat 19 Schüler, drei schreiben die Arbeit nicht mit. Die Arbeit besteht aus fünf Aufgaben mit maximal 6, 4, 8, 3 und 4 Punkten. Die Beschreibung folgt der Reihenfolge der einzelnen Bilder, wie sie auf dem Bildschirm nacheinander erscheinen, beziehungsweise der Zettel, wie sie vom Drukker geschrieben werden.

```
ZAHL DER AUFGABEN        5

AUFGABE  1               6 P
AUFGABE  2               4 P
AUFGABE  3               8 P
AUFGABE  4               3 P
AUFGABE  5               4 P
GESAMT                  25 P
```

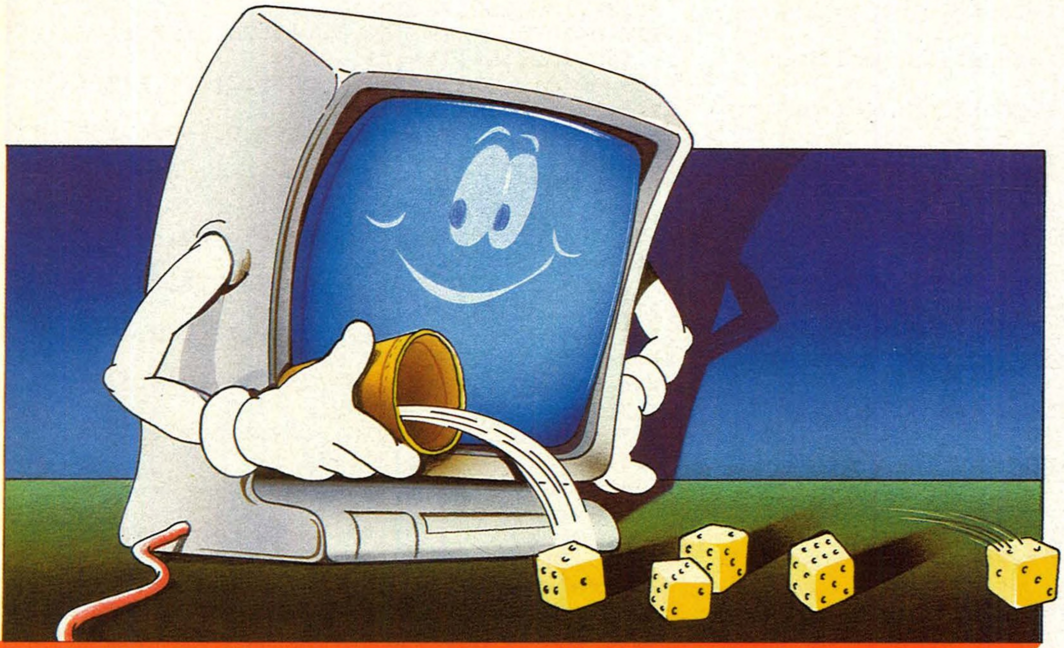
Die Parameter

Die Angabe der maximalen Gesamtpunktzahl durch den Rechner dient als Kontrollmöglichkeit für den Anwender. Bei einer Gesamtpunktzahl kleiner als fünf werden alle Eingaben neu angefordert, weil eine sinnvolle Notenverteilung nicht möglich ist.

Im Normalfall werden später zunächst die Punktzahlen aller Schüler für die erste Aufgabe, dann die Punktzahlen aller Schüler für die zweite Aufgabe und so weiter abgefragt. Es kann aber auch jede Aufgabe mit der vorherigen zu einem Block zusammengefaßt werden. Dann werden die Aufgaben dieses Blockes bei jedem Schüler nacheinander angesteuert. Dazu muß

Scan: S. Höltgen


```
155 DATA174,40,21,76,247,19,202,48,244,1
89,65,21,133,251,189,86
156 DATA21,133,252,76,12,20,169,0,157,44
,21,76,244,19,189,44,21
157 DATA201,1,240,19,173,41,21,157,65,21
,169,16,157,86,21,169,1
158 DATA157,44,21,76,44,20,169,32,160,0,
145,251,169,22,141,42,21
159 DATA169,0,141,43,21,24,189,65,21,109
,42,21,133,251,189,86,21
160 DATA109,43,21,133,252,160,0,177,251,
201,32,208,17,169,81,145
161 DATA251,165,251,157,65,21,165,252,15
7,86,21,76,244,19,201,119
162 DATA240,11,201,120,240,7,201,102,240
,34,76,4,20,169,1,141,42
163 DATA21,165,139,201,127,16,183,56,189
,65,21,237,42,21,133,251
164 DATA189,86,21,237,43,21,133,252,76,7
9,20,169,0,157,44,21,76
165 DATA244,19,173,31,145,41,20,208,5,16
9,2,76,235,20,173,31,145
166 DATA41,24,208,5,169,3,76,235,20,173,
31,145,41,16,208,5,169,4
167 DATA76,235,20,173,31,145,41,4,208,5,
169,5,76,235,20,173,31
168 DATA145,41,8,208,5,169,6,76,235,20,1
69,127,141,34,145,174,32
169 DATA145,169,255,141,34,145,138,41,12
8,208,5,169,7,76,235,20
170 DATA169,1,141,107,21,173,108,21,201,
0,208,29,174,109,21,189
171 DATA132,21,201,1,240,23,141,11,144,2
38,109,21,174,109,21,189
172 DATA132,21,141,108,21,238,109,21,96,
206,108,21,96,169,0,141
173 DATA109,21,96
READY.
```



Fair bleibt der TI-99/4A (mit Extended-BASIC) auch beim Würfeln, obwohl er als Mitspieler unter maximal sechs Teilnehmern leicht das Schicksal zu seinen Gunsten beeinflussen könnt[e]

Berliner Macke

Anfangs würfelt man mit sechs Würfeln gleichzeitig. Nun muß man mindestens einen Würfel, der zählt (beziehungsweise eine zählende Würfelkombination), herauslegen. Das wird im Programm „SETZEN" genannt. Nach dem Setzen kann man aufhören; dann werden die erreichten Punkte dem Konto gutgeschrieben. Man kann aber auch mit den übrigen Würfeln weiterwürfeln. Das Risiko dabei: Hat man nun keinen zählenden Würfel im Wurf, sind alle bisher gesetzten Punkte verfallen, und der nächste Spieler ist an der Reihe. Die Chancen dabei: Man kann sein Punktekonto noch verbessern, und wenn es gelingt, alle sechs Würfel zu setzen, darf man wieder mit allen sechs Würfeln weiterwürfeln.

Wertung

Jede 1 zählt 100 Punkte
Jede 5 zählt 50 Punkte

Alle anderen Zahlen bringen nur dann Punkte, wenn sie in **einem** Wurf öfter als zweimal vorkommen.

1-1-1 zählt 1000 Punkte; jede weitere 1: +1000
2-2-2 zählt 200 Punkte; jede weitere 2: + 200
3-3-3 zählt 300 Punkte; jede weitere 3: + 300
4-4-4 zählt 400 Punkte; jede weitere 4: + 400
5-5-5 zählt 500 Punkte; jede weitere 5: + 500
6-6-6 zählt 600 Punkte; jede weitere 6: + 600
1-2-3-4-5-6 zählt 1000 Punkte (Straße)

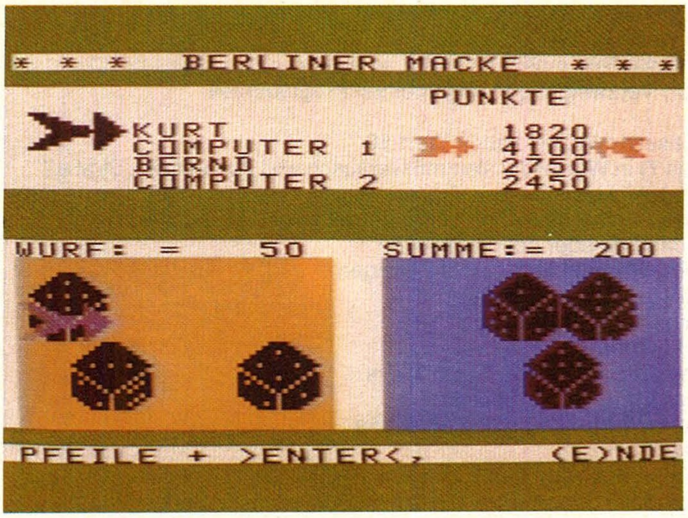
Sieger ist der Spieler, dessen Punktestand als erster 10 000 erreicht oder überschreitet. Die Runde wird aber vorher zu Ende gespielt, damit alle gleiche Chancen haben.

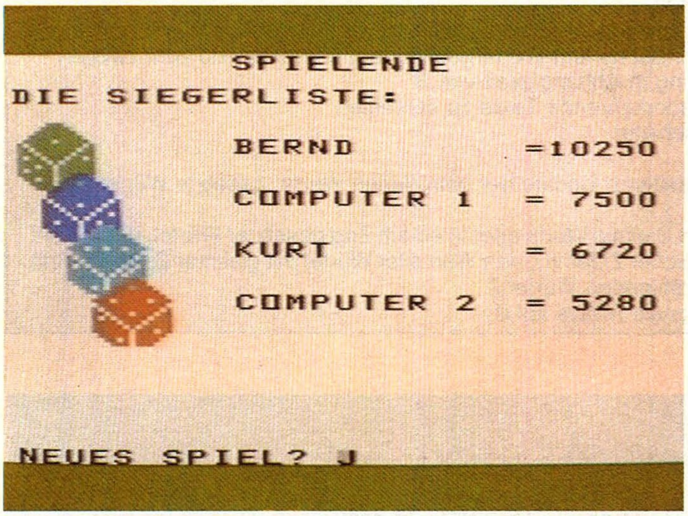
Wer spielt mit?

Nach dem Titelbild kann man die Regeln abrufen, danach verlangt der Computer die Spieleranzahl und die Namen der Mitspieler. Soll der Computer als Spieler aktiv werden, muß man den Buchstaben C als Name eingeben. Der Computer kann sogar alle Spieler ersetzen und spielt dann allein; er wendet immer die gleiche Strategie an.

Nun geht das Spiel los. Im oberen Bildschirm erscheinen der Spielname, die Namensliste und der Punktestand aller Teilnehmer. Ein großer schwarzer Pfeil zeigt an, wer gerade würfeln muß, zwei kleine rote Pfeile zeigen auf die gerade höchste Punktzahl. Im unteren Bildschirmteil erscheinen zwei schwarze Flächen, und in der letzten Zeile, die durch den Programmablauf

Scan: S. Höltgen





führt, blinkt der Name des Spielers, der würfeln soll.

Gewürfelt wird mit der Taste W (wie Würfeln), Geräusch- und Farbeffekte symbolisieren das Schütteln des Würfelbechers, und mit jedem Tastendruck wird das Mischen beendet. Die linke Fläche wird gelb, und die Würfel erscheinen. Über dem Feld wird der Punktwert dieses Wurfs angezeigt. Ein violetter Pfeil zeigt auf den ersten Würfel, der zählt.

Setzen von Würfeln

In der Führungszeile steht, was man jetzt machen kann. Will man nicht mehr weiterwürfeln, drückt man E (wie Ende) und setzt damit alle zählenden Würfel in das rechte blaue Feld. Die Punkte werden dem Konto addiert, die roten Pfeile zeigen auf den nunmehr höchsten Punktstand, und der nächste Spieler darf würfeln. Will man aber nur spezielle Würfel setzen, kann man mit S und D (den Pfeiltasten) den violetten Pfeil verschieben; mit ENTER wird dann der Würfel ins blaue Feld gesetzt, auf den der Pfeil zeigt.

Mit S (wie Setzen) kann man weitere Würfel setzen, mit W weiterwürfeln oder auch mit E doch noch aufhören. Will man eine Zahl setzen, die öfter als zweimal vorkommt, so setzt der Computer automatisch alle diese Würfel. Über dem blauen Feld wird die Summe der gesetzten Würfel angezeigt.

Variable	Bedeutung
A	Anzahl der Spieler
B, C	Zeilen- und Spaltenkoordinaten der Würfel
F	Flag bei Zufallsentscheidung (nein = 1, ja = 2)
G	Anzahl der gesetzten Würfel
I, J, K, L	Zählvariable in Schleifen und Keyroutine und so weiter
M	Höchste Punktzahl aller Spieler
M1	Zweithöchste Punktzahl
M2	Ordnungszahl des Spielers mit der Höchstpunktzahl
N()	Punktwert, den alle Würfel mit gleicher Ziffer ergeben
P()	Punktkonto der Spieler
R	Risikozahl
U.	Punktwert eines Wurfs
U1	Punktwert der Würfel, die in einem Zug gesetzt werden
U2	Punktwert aller gesetzten Würfel
U3	Punktwert aus U1 + U2
V()	Anzahl der Würfel mit gleicher Ziffer
V1()	Anzahl der Würfel mit gleicher Ziffer, die in einem Zug gesetzt werden
W()	Ziffern der Würfel im Würfelfeld
W1()	Ziffern der Würfel im Setzfeld
Z	Anzahl der zählenden Würfel im Wurf
A$	Zwischenspeicher für Text
S$()	Namen der Spieler

Variablenliste

Zeilen	Bedeutung
1– 11	Impressum
130– 470	Initialisierung und Zeichendefinition
170– 180	Definition der Zeilen- und Spaltenkoordinaten der Würfel
210– 300	DATA für Würfelmuster 1 bis 6
220– 250	Scheinbefehle und Prescanning-Abschalten
310– 330	DATA für Pfeilmuster
380– 470	Definieren der Sonderzeichen, Muster und Farbe
510– 540	Titelbild
550– 640	Spielregeln
650– 730	Abfrage Spieleranzahl und Namen
780–2020	Hauptprogramm
780– 910	Setup Bildschirm
920	Verzweigung zum Computerzug
960–1240	Spielerzug
960– 990	Spieleraufruf in Führungszeile
1000–1030	Tastenabfrage, Geräusch und Farbeffekte „Würfeln"
1040	Verzweigung zur Routine „Würfeln"
1050	Verzweigung bei ungültigem Wurf
1060	Verzweigung bei „Straße" und „Alle Würfel gelten"
1070–1240	Behandlung Führungszeile und Tastaturabfrage, Pfeil (Sprite 1) auf ersten Zählwürfel setzen
1280–1530	Gemeinsame Teile für Spieler- und Computerzug
1280–1380	Laufschrift und Trauermusik „Ungültiger Zug"
1390–1490	Punktkonto-Update, Höchstpunktzahl ermitteln, rote Pfeile (Sprite 11 und 12) auf Höchstpunktzahl setzen
1490	zum nächsten Spieler
1530–1710	Endbehandlung
1530	Endetest
1540–1560	Hurra (Spielende)
1580–1610	Sortieren der Spieler nach Punktzahl
1620–1710	Siegerliste schreiben
1750–2020	Computerzug
1760–1790	Effekte „Würfeln" und Verzweigung Routine „Würfeln"
1800	Verzweigung „Ungültiger Zug"
1810	Verzweigung bei „Straße" und „Alle Würfel gelten"

Scan: S. Höltgen

Wenn der Computer spielt, gehen alle diese Schritte automatisch. Automatisch wird auch eine Straße mit Hurra bejubelt, wenn alle Würfel zählen. In diesen Fällen nimmt der Computer an, daß man das ausnutzen will und setzt für den Spieler alle Würfel. Wenn aber das Schicksal zuschlägt und der Wurf keinen zählenden Würfel erhält, wird das mit Trauermusik und Laufschrift begleitet.

Sobald ein Spieler die Grenze von 10 000 Punkten erreicht hat, wird die Runde noch zu Ende gespielt. Dann gibt der Computer die Siegerliste aus, und man kann ein neues Spiel wählen oder aufhören.

Im Listing bedeutet ! ein REM. Alle Zeilen mit ! können weggelassen werden. Die Variablen in Unterprogrammen übernehmen den Wert und die Bedeutung der Variablen in der Übergabeparameterliste. Wo es ging, wurden die Namen aus dem Hauptprogramm beibehalten.

Bernhard Decker

1830–1910	Würfel mit Höchstpunktzahl pro Stück setzen
1920–1950	Zufällig eventuell weiteren Würfel setzen
1960–1980	Verzweigung zur Entscheidungsroutine und so weiter
2000–2020	Computer hört auf; Verzweigung in allgemeinen Programmteil
2100–3520	Subroutinen
2100–2150	Würfeln; schon gesetzte Würfel haben Wert 10
2170–2280	Würfel (Sprite 2 bis 7) in Wurffeld malen mit Geräusch und zu Routine „Straße?" und „Alle gelten" schicken
2320–2480	Pfeile durch Tastatur verschieben, geltende Würfel eventuell setzen
2460	Kontrolle Würfel ungültig oder schon gesetzt?
2520–2600	Würfel gleitend von Wurffeld in Setzfeld bewegen
2540–2580	Automatisches Setzen bei Mehrlingen
2640–2690	Update-Setzsumme, Wurffeld löschen
2730–2800	Hurra (alle Würfel gelten)
2840–2930	Test Straße (keine Ziffer ist doppelt) und Hurra
2970–3060	Würfel springen von Wurffeld in Setzfeld
3100–3140	sucht alle geltenden Würfel und ruft Gleitroutine
3190–3240	Laufschrift „Macke"
3280–3370	Entscheidung „Aufhören"; ermittelt Risikozahl (5<R<96)
3330	Formel, wenn Computer nicht führt. R wird größer, je mehr übrige Würfel, je näher Höchstpunktzahl dem Ziel, bei kleiner Setzsumme und je größer der Abstand zum ersten
3360	Formel, wenn Computer führt. R wird kleiner, je weniger übrige Würfel, je kleiner Abstand zum Ziel, bei hohem Wurfergebnis, je größer Abstand zum zweiten
3410–3420	Zufallsentscheidung in Abhängigkeit von R
3460–3520	Routinen, um wiederkehrende Texte zu schreiben
3570	Prescanning einschalten
3580–3890	Unterprogramme
3580–3660	Bewegt Sprites gleitend; Update der betroffenen Werte; gesetzte Würfel erhalten den Wert 10
3680–3850	Ermittelt den Wert U eines Wurfs oder in einem Zug gesetzter Würfel, die Anzahl der Würfel mit gleicher Ziffer v(), den Wert aller Würfel mit gleicher Ziffer N() und die Anzahl aller zählenden Würfel Z
3870–3890	Setzt eindimensionale Arrays auf 0

Programmstruktur

```
110 ! INITIALISIERUNG
130 RANDOMIZE
140 CALL MAGNIFY(4)
150 OPTION BASE 1
160 DIM S$(6),P(6),W(6),W1(6),V(6),V1(6)
,N(6)
170 DEF B(X)=(13+INT((X-1)/3)*4)*8
180 DEF C(X)=(-1+3*X-INT((X-1)/3)*7.5)*9
190 IMAGE =#####
200 !
210 DATA 0103070F1F2F373B,2D3E3B2F3F1B0F
07,C0E0F0F87CFAF6EE,DABE7E7E7E6C7870
220 GOTO 250 :: I,J,K,L,A,F,G,M,M1,M2,R,
U,U1,U2,U3,Z,A$
230 CALL CLEAR :: CALL KEY :: CALL SOUND
 :: CALL CHAR :: CALL HCHAR :: CALL COLO
R :: CALL SCREEN :: CALL SPRITE :: CALL
LOCATE :: CALL MOTION :: CALL DELSPRITE
240 CALL MOV :: CALL WERT :: CALL NULL
250 !@P-
260 DATA 0103070F1F1F2F37,3B3D3E373F1F0F
07,C0E070F8FCFCFA76,EEDABE6E7A7C6870
270 DATA 0103070F1F3B1F2F,372B3D3E3F1F0D
07,C0E0F0F8FC6EFCFA,F6EAD6AE7A746870
280 DATA 0103070F1F3B1F2F,372B3D3E3F1F0D
07,C0E070F8FCEEFCFA,76EEDEBE767C7870
290 DATA 0103070F1F3B1F2F,373B3D3E371F0F
07,C0E070F8FC6EFCFA,76ECDEBE767C7850
300 DATA 0103070F1E3F1B2F,363B2D3E3F1F0D
07,C0E0B0F8ECFEBCFA,F6EADEBE5A7C7850
310 DATA 00000000000000E0,783E1F0F1F3E78
E0,0000000000000010,181CDEFFDE1C1810
320 DATA 0000000000000000,00236FFF6F2300
00,0000000000000000,00C0800080C00000
330 DATA 0000000000000000,00030100010300
00,0000000000000000,00C4F6FFF6C40000
340 CALL CLEAR
350 !
360 ! DEF
380 FOR I=1 TO 36
390 READ A$
400 CALL CHAR(95+I,A$)
410 NEXT I
420 CALL CHAR(132,"FFFFFFFFFFFFFFFF",133
,"",136,"")
430 !
440 CALL SCREEN(13)
450 CALL COLOR(14,2,13)
460 CALL HCHAR(1,1,136,768)
470 FOR I=0 TO 8 :: CALL COLOR(I,2,16)::
 NEXT I :: CALL CLEAR
480 !
490 ! TITEL
510 FOR I=1 TO 6 :: CALL SOUND(100,-7,4)
:: CALL SPRITE(#(8-I),92+4*I,2+2*I,10+24
*I,7+8*I):: DISPLAY AT(3+3*I,5+I):"BERLI
NER MACKE" :: NEXT I
520 DISPLAY AT(24,12):"REGELN? J"
530 ACCEPT AT(24,20)VALIDATE("JN")SIZE(-
1):A$
540 CALL DELSPRITE(ALL):: CALL CLEAR ::
IF A$="N" THEN 650
550 PRINT "BERLINER MACKE": : :"MAN HAT
6 WUERFEL UND WIRFT ALLE GLEICHZEITIG.":
 :
560 PRINT "NACH JEDEM WURF MINDESTENS  1
 ZAEHLENDEN WUERFEL SETZEN": :"ES ZAEHLE
N: JEDE 1 = 100":TAB(13);"JEDE 5 =  50":
 :
570 PRINT "ANDERE ZIFFERN NUR,WENN SIE O
EFTER ALS 2-MAL AUFTRETEN.": :"1-1-1 =10
00; +JEDE 1:+1000":"2-2-2 = 200, +JEDE 2
: +200"
580 PRINT "3-3-3 = 300, +JEDE 3: +300":"
4-4-4 = 400, +JEDE 4: +400"
590 PRINT "5-5-5 = 500, +JEDE 5: +500":"
6-6-6 = 600, +JEDE 6: +600": : :: GOSUB
3500
600 PRINT "AUSSERDEM ZAEHLT NOCH DIE   S
TRASSE:": :"1-2-3-4-5-6 = 1000": : :
610 PRINT "GESETZTE WUERFEL SIND TOT,  S
IND ALLE GESETZT, DARF MAN MIT ALLEN WEI
TERWUERFELN": :
620 PRINT "NUR NACH DEM SETZEN KANN MANA
UFHOEREN UND PUNKTE SCHREI-BEN.": :"HAT
MAN VORHER EINEN UNGUEL-TIGEN WURF, SO V
ERFALLEN"
630 PRINT "DIE GESETZTEN WUERFEL.": : :
:: GOSUB 3500
640 PRINT "SPIELENDE BEI 10000 PUNKTEN":
 : : :
650 PRINT "UEBRIGENS:": :"DER COMPUTER W
ILL SIE GERNE BESIEGEN.": :"SEIN NAME: >
```

Scan: S. Höltgen


```
C<": : :"SPIELERANZAHL (2 BIS 6): 2": :
660 ACCEPT AT(22,26)VALIDATE("23456")SIZ
E(-1):A
670 PRINT "DIE SPIELERNAMEN:": :
680 M2,J=1 :: FOR I=1 TO A
690 PRINT "SPIELER ";STR$(I);
700 INPUT ": ":S$(I)
710 IF S$(I)<>"C" THEN 730
720 S$(I)="COMPUTER "&STR$(J):: J=J+1
730 NEXT I :: CALL HCHAR(1,1,136,768)
740 !
750 ! HAUPTPRGRM
760 ! **********
780 FOR J=1 TO A
790 CALL NULL(W()):: CALL DELSPRITE(ALL)
:: CALL HCHAR(1,1,136,768)
800 G,U2=0
810 GOSUB 3200
820 CALL HCHAR(3,1,32,32*(A+2))
830 DISPLAY AT(3,18):"PUNKTE:"
840 FOR I=1 TO A
850 DISPLAY AT(4+I,6):S$(I)
860 DISPLAY AT(4+I,20):USING "#####":P(I
)
870 NEXT I
880 CALL SPRITE(#11,124,10,6+8*M2,208,#1
2,128,10,6+8*M2,136)
890 CALL COLOR(13,16,16):: CALL HCHAR(12
,1,32,352)
900 FOR I=13 TO 22 :: CALL HCHAR(I,3,132
,13):: CALL HCHAR(I,18,133,13):: NEXT I
:: CALL COLOR(13,2,2)
910 CALL SPRITE(#10,120,2,7+8*J,23)
920 IF SEG$(S$(J),1,4)="COMP" THEN 1750
930 !
940 ! SPIELERZUG
960 CALL SOUND(100,880,5)
970 CALL KEY(0,K,L)
980 DISPLAY AT(24,1):S$(J);" (W)UERFELN!
" :: CALL HCHAR(24,1,32,32)
990 IF L=0 THEN 970
1000 DISPLAY AT(24,1):"TASTE DRUECKEN!"
:: CALL HCHAR(12,1,32,16)
1010 CALL SOUND(150,-INT(RND*3+5),3):: C
ALL COLOR(13,16,16)
1020 CALL KEY(0,K,L):: CALL COLOR(13,2,6
)
1030 IF L=0 THEN 1010
1040 GOSUB 2100
1050 IF Z=0 THEN 1280
1060 IF Z>6 THEN GOSUB 2970 :: GOSUB 348
0 :: GOTO 1130
1070 DISPLAY AT(24,1):"PFEILE + >ENTER<,
 (E)NDE"
1080 FOR I=1 TO 6 :: IF W(I)=10 THEN 110
0
1090 IF V(W(I))>0 THEN 1110
1100 NEXT I
1110 CALL SPRITE(#1,120,14,B(I),C(I))
1120 GOSUB 2330 :: GOSUB 3480 :: IF Z>0
OR K=69 THEN 1180
1130 CALL KEY(0,K,L)
1140 DISPLAY AT(24,11):"(W)UERFELN  (E)N
DE"
1150 IF K=87 THEN 1220
1160 IF K=69 THEN 1230
1170 CALL HCHAR(24,1,32,32):: GOTO 1130
1180 IF K=69 THEN 1390
1190 CALL KEY(0,K,L)
1200 DISPLAY AT(24,1):"(S)ETZEN  (W)UERF
ELN  (E)NDE"
1210 IF K=83 THEN 1070
1220 IF K=87 THEN GOSUB 2640 :: GOTO 100
0
1230 IF K=69 THEN GOSUB 3100 :: GOTO 139
0
1240 CALL HCHAR(24,1,32,32):: GOTO 1190
1250 !
1260 ! WRF UNGLTG
1270 !
1280 A$="LEIDER GAR NICHTS!" :: GOSUB 31
90 :: FOR I=1 TO 10
1290 L=INT(RND*50)+125 :: CALL SOUND(-40
0,L,1,L+10,1,L+40,3,-INT(RND*3+5),2)
1300 CALL SCREEN(12-I):: CALL COLOR(14,2
,2)
1310 DISPLAY AT(24,23-2*I)SIZE(4+2*I):A$
1320 CALL SOUND(-400,-INT(RND*3+5),12)
1330 CALL COLOR(14,2,3)
1340 NEXT I
1350 DISPLAY AT(24,1):A$
1360 FOR I=6000 TO 600 STEP -200 :: CALL
 SOUND(-2000,I,4):: NEXT I
1370 CALL SOUND(200,150,0,152,0,154,0)
1380 CALL SCREEN(3):: GOTO 1490
1390 U2=U2+U1
1400 P(J)=P(J)+U2 :: DISPLAY AT(4+J,20):
USING "#####":P(J)
1410 M=0 :: FOR I=1 TO 6 :: IF P(I)>M TH
EN M1=M :: M=P(I):: M2=I
1420 NEXT I
1430 M1=0 :: FOR I=1 TO 6 :: IF I=M2 OR
P(I)<M1 THEN 1440 ELSE M1=P(I)
1440 NEXT I
1450 CALL SPRITE(#11,124,1,6+8*M2,208,#1
2,128,1,6+8*M2,136)
1460 FOR I=1 TO 4 :: CALL SOUND(200,880,
6):: CALL COLOR(#11,1,#12,10)
1470 CALL SOUND(200,880,6):: CALL COLOR(
#11,10,#12,1):: NEXT I
1480 CALL SOUND(1000,987,3,830,4,659,5):
: CALL COLOR(#12,10):: CALL SOUND(1000,1
108,4,880,5)
1490 NEXT J
1500 !
1510 ! ENDE?
1520 !
1530 IF M<10000 THEN 780
1540 CALL DELSPRITE(ALL)
1550 CALL HCHAR(1,1,136,768):: FOR I=1 T
O 10 :: CALL SOUND(-150,830,3,659,4,293,
4):: CALL SCREEN(14-I)
1560 CALL SOUND(1000,880,2,659,4,277,4):
: NEXT I
1570 CALL SCREEN(13):: CALL CLEAR :: DIS
PLAY AT(1,10):"SPIELENDE"
1580 FOR I=1 TO 5 :: FOR J=I+1 TO 6
1590 IF P(I)>=P(J)THEN 1610
1600 M=P(I):: A$=S$(I):: P(I)=P(J):: S$(
I)=S$(J):: P(J)=M :: S$(J)=A$
1610 NEXT J :: NEXT I
1620 DISPLAY AT(3,1):"DIE SIEGERLISTE:"
1630 FOR I=1 TO A
1640 CALL SOUND(100,-7,4)
1650 CALL SPRITE(#(8-I),92+4*I,2+2*I,10+
24*I,7+8*I)
1660 DISPLAY AT(3+3*I,10):S$(I)
1670 DISPLAY AT(3+3*I,22):USING 190:P(I)
1680 NEXT I
1690 DISPLAY AT(24,1):"NEUES SPIEL? J"
1700 ACCEPT AT(24,14)VALIDATE("JN")SIZE(
-1):A$
1710 CALL DELSPRITE(ALL):: CALL HCHAR(1,
1,136,768):: IF A$="J" THEN RUN 140 ELSE
STOP
1720 !
1730 ! COMP.ZUG
1740 !
1750 U2=0
1760 DISPLAY AT(24,1)BEEP:S$(J);" WUERFE
LT" :: CALL HCHAR(12,1,32,16)
1770 K=INT(RND*15)+1
1780 FOR I=1 TO K :: CALL COLOR(13,2,6):
: CALL SOUND(150,-INT(RND*3+5),3):: CALL
 COLOR(13,16,16):: NEXT I
1790 CALL NULL(N()):: GOSUB 2100
1800 IF Z=0 THEN 1280
1810 IF Z>5 THEN GOSUB 2970 :: U3=U+U2 :
: GOSUB 3280 :: ON F GOTO 2000,1970
1820 ! WRFLWAHL
1830 GOSUB 3460
1840 K=-10 :: FOR I=1 TO 6
1850 IF V(I)=0 THEN 1880
1860 IF INT(N(I)/V(I)+.5)<=K THEN 1880
1870 K=N(I):: L=I
1880 NEXT I
1890 FOR I=1 TO 6 :: IF W(I)=L THEN 1910
1900 NEXT I
1910 GOSUB 2520
1920 CALL NULL(N()):: IF Z=0 THEN 1960
1930 R=10 :: GOSUB 3410
1940 IF F=1 THEN 1960
1950 CALL NULL(V()):: CALL WERT(V(),W(),
U,Z,N()):: GOTO 1840
1960 U3=U2+U1 :: GOSUB 3280 :: ON F GOTO
 1980,1970
1970 GOSUB 2640 :: GOTO 1760
1980 GOSUB 3100
1990 ! STOP
2000 GOSUB 3480
```

Scan: S. Höltgen


```
2010 DISPLAY AT(24,1)BEEP:S$(J);" HOERT
AUF"
2020 GOTO 1390
2030 !
2040 ! SUBROUT
2050 ! *******
2060 !
2070 !
2080 ! WRFELN
2090 !
2100 CALL COLOR(13,11,6):: CALL NULL(W1(
))
2110 FOR I=1 TO 6
2120 IF W(I)=10 THEN 2140
2130 W(I)=INT(RND*6)+1
2140 NEXT I
2150 CALL HCHAR(24,1,32,32)
2160 ! MALT W.
2170 DISPLAY AT(12,1)SIZE(21):"WURF:
   SUMME:" :: DISPLAY AT(12,22):USING
 190:U2
2180 FOR I=1 TO 6
2190 IF W(I)=10 THEN 2220
2200 CALL SPRITE(#(I+1),92+4*W(I),2,B(I)
,C(I))
2210 CALL SOUND(50,-5,0):: CALL SOUND(50
,-6,3):: CALL SOUND(200,-7,7)
2220 NEXT I
2230 IF G>0 THEN 2250 ELSE GOSUB 2840
2240 IF Z=10 THEN 2270
2250 CALL NULL(V()):: CALL WERT(V(),W(),
U,Z,N())
2260 IF Z+G=6 THEN Z=20 :: GOSUB 2730
2270 DISPLAY AT(12,7)SIZE(6):USING 190:U
2280 RETURN
2290 !
2300 ! PFEIL
2310 !
2320 CALL SOUND(50,200,7,205,7,209,7)
2330 CALL KEY(0,K,L)
2340 CALL COLOR(#1,1)
2350 IF K=68 THEN I=MIN(I+1,6):: GOTO 24
10
2360 IF K=83 THEN I=MAX(I-1,1):: GOTO 24
10
2370 CALL COLOR(#1,14)
2380 IF K=69 THEN GOSUB 3100 :: RETURN
2390 IF K=13 THEN 2430
2400 GOTO 2330
2410 CALL COLOR(#1,14):: CALL LOCATE(#1,
B(I),C(I)):: GOTO 2320
2420 ! WRFL SETZEN
2430 CALL LOCATE(#1,B(I),C(I))
2440 CALL COINC(#1,#(I+1),10,K)
2450 IF K=0 THEN 2330
2460 IF V(W(I))=0 OR W(I)=10 THEN 2330
2470 GOSUB 2520
2480 RETURN
2490 !
2500 ! W. GLEITEN
2510 !
2520 CALL SPRITE(#1,120,14,B(I),C(I))::
CALL MOV(I,B(I),C(I),W1(I),W(I),G,Z,V())
2530 IF (W1(I)=1 AND V(1)<2)OR(W1(I)=5 A
ND V(5)<2)THEN 2590
2540 FOR L=1 TO 6
2550 IF W(L)<>W1(I)THEN 2580
2560 CALL LOCATE(#1,B(L),C(L))
2570 CALL MOV(L,B(L),C(L),W1(L),W(L),G,Z
,V())
2580 NEXT L
2590 CALL DELSPRITE(#1):: CALL NULL(V1()
):: CALL WERT(V1(),W1(),U1,L,N())
2600 GOSUB 3480 :: RETURN
2610 !
2620 ! CLEAR WRFFLD
2630 !
2640 U2=U2+U1
2650 FOR I=1 TO 6 :: IF W(I)=10 THEN 267
0
2660 CALL DELSPRITE(#(I+1))
2670 NEXT I
2680 CALL COLOR(13,2,6)
2690 RETURN
2700 !
2710 ! ALLE GELTEN
2720 !
2730 FOR I=1 TO 6
2740 CALL SOUND(100,700,3,710,3,730,3)::
 CALL HCHAR(24,1,32,32)
2750 CALL COLOR(13,2,2)
2760 CALL SOUND(-400,900,3,910,3,930,3)
2770 CALL COLOR(13,11,6)
2780 DISPLAY AT(24,1):"ALLE WUERFEL GELT
EN!"
2790 NEXT I
2800 U1=U :: RETURN
2810 !
2820 ! STRASSE?
2830 !
2840 FOR I=1 TO 5 :: FOR K=I+1 TO 6
2850 IF W(I)=W(K)THEN 2930
2860 NEXT K :: NEXT I
2870 U,U1=1000 :: Z=10
2880 FOR I=1 TO 15
2890 CALL SOUND(-400,1200,3,1220,3,1290,
3):: CALL MAGNIFY(3)
2900 CALL SOUND(-400,800,3,820,3,890,3):
: CALL MAGNIFY(4)
2910 NEXT I
2920 FOR I=1 TO 10 :: CALL SOUND(-400,I*
600,2):: DISPLAY AT(24,22-2*I):"STRASSE"
 :: NEXT I
2930 RETURN
2940 !
2950 ! WRFL JUMP
2960 !
2970 FOR I=1 TO 6
2980 IF W(I)=10 THEN 3040
2990 CALL SOUND(-100,-7,3)
3000 CALL SPRITE(#1,120,14,B(I),C(I))
3010 CALL SOUND(300,1100,4,1300,4)
3020 CALL LOCATE(#(I+1),B(I),C(I)+122)
3030 W1(I)=W(I)
3040 NEXT I
3050 G,Z=0 :: CALL NULL(W()):: CALL DELS
PRITE(#1)
3060 GOSUB 3480 :: RETURN
3070 !
3080 ! SETZT ALLE GUELT W.
3090 !
3100 IF Z=0 THEN 3150
3110 FOR I=1 TO 6
3120 IF W(I)>6 THEN 3140
3130 IF V(W(I))>0 THEN GOSUB 2520
3140 NEXT I
3150 RETURN
3160 !
3170 ! MACKE
3180 !
3190 CALL HCHAR(1,1,136,32)
3200 FOR I=1 TO 10
3210 CALL SOUND(-300,210+10*I,7)
3220 DISPLAY AT(1,21-2*I)SIZE(8+2*I):"*
* *  BERLINER MACKE  * * *"
3230 NEXT I
3240 RETURN
3250 !
3260 ! STOPPEN?
3270 !
3280 K=P(J)+U3 :: IF K<M THEN 3320
3290 IF P(J)<M THEN M1=M
3300 GOTO 3360
3310 !CMP=HINTEN
3320 R=(7-G)*15+8-(10000-MAX(1000,M))/10
00-U3/100+(M-K)/1000
3330 GOTO 3370
3340 !CMP FUEHRT
3350 IF K>=10000 THEN R=0 :: RETURN
3360 R=(6-G)*15+(10000-K)/2000-U3/80+10-
(K-M1)/1000
3370 R=MAX(5,MIN(R,95)):: GOSUB 3410 ::
RETURN
3380 !
3390 ! ZUFLENTSCHDG
3400 !
3410 IF INT(RND*100)+1<=R THEN F=2 ELSE
F=1
3420 RETURN
3430 !
3440 ! TEXT
3450 !
3460 DISPLAY AT(24,1)BEEP:S$(J);" SETZT"
 :: RETURN
3470 ! SUMME
3480 DISPLAY AT(12,22):USING 190:U2+U1 :
: RETURN
3490 ! WEITER
3500 DISPLAY AT(24,22):">ENTER<"
3510 CALL KEY(0,K,L):: IF L=0 THEN 3510
```

```
3520 CALL CLEAR :: RETURN
3530 !
3540 ! UNTERPRGRM
3550 ! **********
3560 !
3570 !@P+
3580 SUB MOV(I,B,C,W1,W,G,Z,V())
3590 CALL MOTION(#(I+1),0,17)
3600 J=(RND*6+2)*75
3610 CALL SOUND(-500,J,7,J+10,7,J+30,12)
3620 CALL COINC(#(I+1),B,C+122,9,K)
3630 IF K=0 THEN 3610
3640 CALL MOTION(#(I+1),0,0):: CALL LOCA
TE(#(I+1),B,C+122)
3650 V(W)=V(W)-1 :: W1=W :: W=10 :: G=G+
1 :: Z=Z-1
3660 SUBEND
3670 !
3680 SUB WERT(V(),W(),U,Z,N())
3690 Z=0 :: FOR I=1 TO 6
3700 IF W(I)=10 OR W(I)=0 THEN 3720
3710 V(W(I))=V(W(I))+1
3720 NEXT I
3730 U=0
3740 IF V(1)>2 THEN U=1000+(V(1)-3)*1000
 ELSE U=V(1)*100
3750 N(1)=U :: Z=Z+V(1)
3760 FOR I=2 TO 6
3770 IF I<>5 THEN 3800
3780 Z=Z+V(5)
3790 IF V(5)>2 THEN 3820 ELSE N(5)=V(5)*
50 :: GOTO 3830
3800 IF V(I)<3 THEN V(I),N(I)=0 :: GOTO
3840
3810 Z=Z+V(I)
3820 N(I)=I*100+(V(I)-3)*I*100
3830 U=U+N(I)
3840 NEXT I
3850 SUBEND
3860 !
3870 SUB NULL(V())
3880 FOR I=1 TO 6 :: V(I)=0 :: NEXT I
3890 SUBEND
```

Snake

Groß und stark will eine Schlange auf dem Schneider CPC werden und verspeist ein Kaninchen nach dem anderen, aber es gibt auch Igel

Der Versuch, einen Igel zu fressen, kann für die Schlange fatal enden. Außerdem muß sie sich davor hüten, in die Umrandung zu beißen oder sich selbst anzuknabbern.

Das Spiel kann mit ein oder zwei Spielern gespielt werden, bei zwei Spielern tritt dann noch eine gegnerische Schlange als Hindernis auf. Die Steuerung der Schlangen wird für Spieler 1 und Spieler 2 über folgende Tasten kontrolliert.

Spieler 1:	Spieler 2:
4 links	a links
6 rechts	s rechts
8 hoch	w hoch
2 runter	z runter

Spieler 1 steuert die Schlange mit den ausgefüllten Segmenten, Spieler 2 die andere.

Das Programm nutzt die hervorragenden Ton- und Grafikmöglichkeiten des Schneider aus und zeigt, daß man auch in BASIC relativ schnelle Spiele programmieren kann.

Thomas Burg

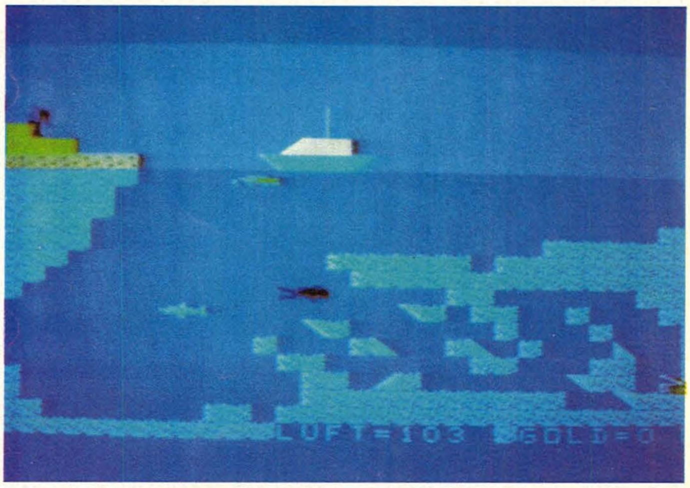
```
160 DEFINT a-z: MODE 1: INK 0,0:
INK 1,3: INK 2,6: INK 3,24: BORDE
R 0
170 REM *** Definieren der einzel
nen Zeichen
180 SYMBOL 240,0,&3C,&66,&42,&42,
&66,&3C,0:          REM *** Schlange
nsegment 1
190 SYMBOL 241,0,&3C,&7E,&7E,&7E,
&7E,&3C,0:          REM *** Schlange
nsegment 2
200 SYMBOL 242,&24,&24,&7E,&DB,&9
9,&FF,&7E,&18:      REM *** Schlange
nkopf hoch
210 SYMBOL 243,&38,&6C,&67,&FC,&F
C,&67,&6C,&38:      REM ***        "
     rechts
220 SYMBOL 244,&18,&7E,&FF,&99,&D
B,&7E,&24,&24:      REM ***        "
     runter
230 SYMBOL 245,&1C,&36,&E6,&3F,&3
F,&E6,&36,&1C:      REM ***        "
     links
240 SYMBOL 246,9,5,5,7,&7F,&FF,&F
F,&76:              REM *** Kaninche
n
250 SYMBOL 247,0,&14,&5D,&FE,&7F,
&FF,&7E,&55:        REM *** Igel
260 SYMBOL 248,&99,&5A,&3C,&FF,&F
F,&3C,&5A,&99:      REM *** Wand
270 ENT 2,2,0,1,2,-1,1,2,-2,1,2,-
4,1,2,-8,1
280 ENT 3,50,-1,1
290 ENT 1,20,5,1,20,-5,1
300 FOR i=1 TO 4: READ xr(i),yr(i
): NEXT i
```


Scan: S. Höltgen

Scan: S. Höltgen

Der verschollene Schatz

Auf einer Tauchexpedition haben Sie tief im Meer einen Schatz gefunden. Die 20 Goldbarren aus der Grotte zu bergen, ist auf dem TI-99/4A in der Grundversion mit vielen Gefahren verbunden



Im Meer lauern Krokodile und Haie, die Sie bei Berührung auffressen. Die Grotte kann durch einen Steinschlag zugeschüttet werden, Ihre Luft kann ausgehen, oder Sie werden bewußtlos. Wenn Sie aufgefressen wurden oder wenn die Grotte zugeschüttet ist, hört das Spiel auf. Bewußtlos werden Sie, wenn Sie gegen einen Fels schwimmen. Dann ist Ihre Bewegung für kurze Zeit gesperrt.

Achtung, Steinschlag

Das Erscheinen von Krokodil und Hai und das Vorkommen von Steinschlag werden vom Zufall bestimmt, für letzteres ist die Wahrscheinlichkeit sehr gering. Zeilen 2120 und 2340 sorgen dafür, daß kein neues Krokodil (Hai) erzeugt wird, wenn bereits ein Krokodil (Hai) unterwegs ist. Sie haben nur eine bestimmte Menge Luft zur Verfügung. Wenn die Luft verbraucht ist, hört das Spiel auf. Bekanntlich sind Spiele in TI-BASIC nicht sehr schnell, das Spiel erhält die Spannung jedoch dadurch, daß durch Erscheinen von Hai oder Krokodil das Auftauchen verzögert und die Luft knapp werden kann.

Beim Erreichen der Schatztruhe müssen Sie die Goldbarren herausnehmen und damit zum Schiff zurückschwimmen. Wenn Sie am Schiff angekommen sind, laden Sie das Gold ab und tanken außerdem Sauerstoff. Um die Einzelheiten brauchen Sie sich jedoch nicht weiter zu kümmern, denn es geschieht automatisch, wenn Sie sich direkt unter dem Schiff befinden.

Schrittweise Bergung

Gold nimmt man, indem man über der Schatztruhe die Taste mit dem Pfeil nach unten drückt. Sie können aber nicht mehr als fünf Goldbarren auf einmal nehmen. Und Vorsicht! Bei mehr als drei Goldbarren verbrauchen Sie doppelt soviel Luft. Tip: Mußte man schon auf dem Hinweg zur Grotte Verzögerungen in Kauf nehmen, so empfiehlt es sich, nicht mehr als drei Barren zu nehmen.

Während des Spiels werden in der untersten Zeile der Luftvorrat und das aus der Truhe entnommene Gold eingeblendet, zusätzlich erscheinen (akustisch unterstützt) Gefahrenmeldungen. Man kann das Spiel vorzeitig abbrechen.

Am Schluß des Spiels erscheinen ein Kommentar und die Gesamtmenge der gehobenen Barren.

Steuertasten: E, X, S D (in Pfeilrichtung)

Aufgeben: Taste Q

Während des Spiels muß die Alpha-Lock-Taste gedrückt sein. *Michael Scheibel*

Nummer	Figur
33–36	Fels
37, 38	Hai
39, 40, 41	
43, 120, 121	Boot
60, 152, 88	Insel
96, 97	Taucher
100	Lawine
104, 105	Krokodil
112	Schatz
128	Himmel
136	Leerzeichen
144, 145	Palme

Selbstdefinierte Zeichen

Zeilen	Bedeutung
80– 410	Titelbild
420– 560	Titelmelodie
570– 880	Anleitung
890–1160	Zeichendefinition
1170–1330	Farben
1340–1590	Bildaufbau
1600–1990	Hauptspielroutine
2000–2070	Tastenabfrage
2120–2330	Erscheinen und Bewegung von Krokodilen
2340–2610	Erscheinen und Bewegung von Haien
2620–2720	Steinschlag
2730–2930	CALL-GCHAR-Abfrage
2990–3040	Printroutine
3050–3200	Bewußtlosigkeit
3240–3330	Bewegung des Tauchers
3440–3900	Schlußtexte

Programmstruktur

```
10 REM  ******************
20 REM   DER VERSCHOLLENE                SCH
ATZ
30 REM  ******************
40 REM  PROGRAMM BY M. SCHEIBEL
50 REM  UND R. SCHEIBEL
60 REM
70 REM  TI-BASIC
80 CALL CLEAR
90 CALL SCREEN(2)
100 CALL COLOR(9,13,2)
110 FOR I=1 TO 8
120 CALL COLOR(I,5,2)
130 NEXT I
140 CALL CHAR(98,"3C4299A1A199423C")
150 CALL CHAR(97,"3E6B3E2A4A928989")
160 PRINT "         aaaa  a aa a    "
170 PRINT "        a    aa a    aa "
180 PRINT "       a  DER          a "
190 PRINT "      aa                a"
200 PRINT "      a  VERSCHOLLENE  a"
210 PRINT "      a                a "
220 PRINT "       a       SCHATZ a "
230 PRINT "         a  a         a "
240 PRINT "          aa  aaaaaaa    "
250 PRINT
260 PRINT
270 PRINT
280 PRINT
290 PRINT
300 PRINT
310 PRINT
320 PRINT
330 PRINT
340 PRINT "       b BY M. SCHEIBEL"
```

Scan: S. Höltgen

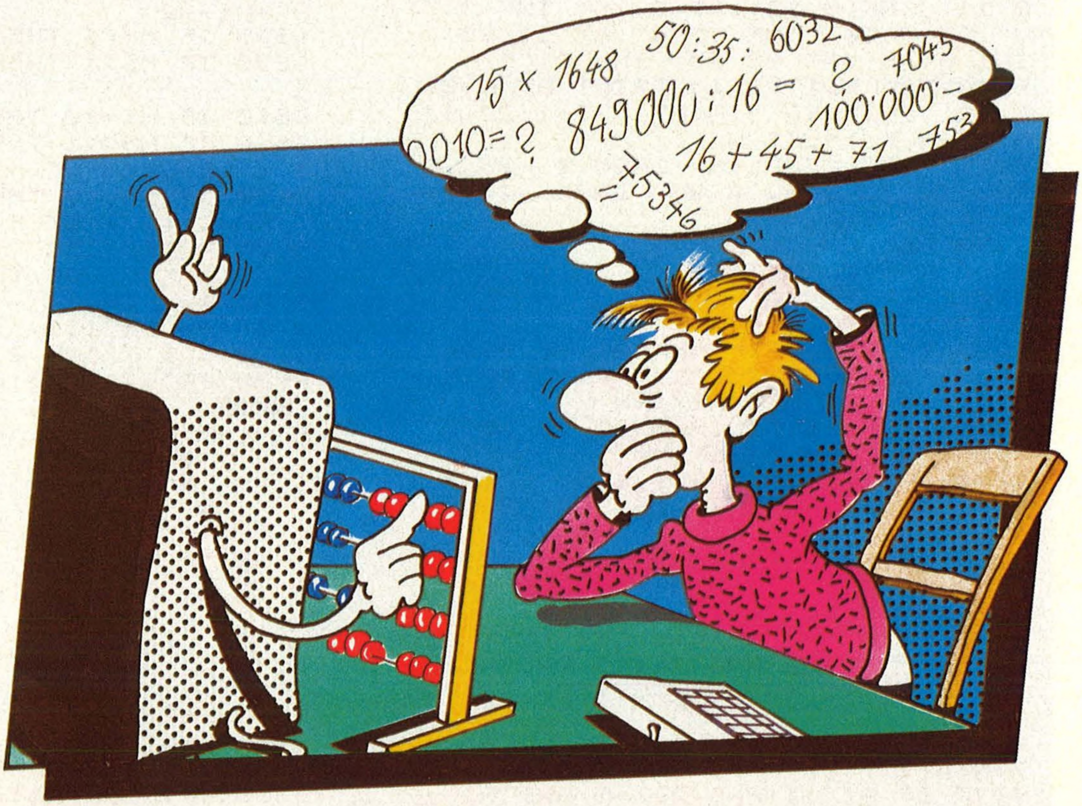

```
350 PRINT
360 CALL CHAR(159,"183C7EFFFF7E3C18")
370 CALL HCHAR(1,1,159,32)
380 CALL HCHAR(24,1,159,32)
390 CALL VCHAR(1,1,159,24)
400 CALL VCHAR(1,32,159,24)
410 CALL COLOR(16,9,2)
420 FOR M=1 TO 2
430 GOSUB 490
440 CALL SOUND(700,659,0,523,0,391,0)
450 GOSUB 490
460 CALL SOUND(700,523,0,391,0,329,0)
470 NEXT M
480 GOTO 550
490 CALL SOUND(900,587,0,466,0,391,0)
500 FOR N=1 TO 2
510 CALL SOUND(200,587,0,440,0,349,0)
520 NEXT N
530 CALL SOUND(600,587,0,466,0,391,0)
540 RETURN
550 FOR I=1 TO 200
560 NEXT I
570 CALL CHAR(159,"FFFFFFFFFFFFFFFF")
580 CALL COLOR(16,5,5)
590 CALL SCREEN(5)
600 CALL HCHAR(1,1,159,768)
610 FOR I=1 TO 12
620 CALL COLOR(I,16,5)
630 NEXT I
640 INPUT "  SPIELANLEITUNG(J/N)?":L$
650 IF L$="N" THEN 890
660 CALL CLEAR
670 PRINT "SIE SIND EIN HOBBYTAUCHER    U
ND HABEN EINEN VERSCHOL-    LENEN SCHATZ
MIT 20 GOLDBAR-REN IN EINER "
680 PRINT "GROTTE UNTER DEM MEER GE-    F
UNDEN. NUN WOLLEN SIE DEN  SCHATZ BERGEN
. DOCH LAUERT  GEFAHR. ES"
690 PRINT "KOENNEN KROKODILE UND HAIE  K
OMMEN, DIE GROTTE KANN      DURCH EINEN S
TEINSCHLAG ZU- GESCHUETTET"
700 PRINT "WERDEN, IHRE LUFT KANN AUS- G
EHEN ODER SIE WERDEN BE-    WUSSTLOS, WEI
L SIE GEGEN    EINEN FELS GE-"
710 PRINT "SCHWOMMEN SIND."
720 PRINT
730 PRINT "   WEITER MIT SPACE-TASTE"
740 CALL KEY(0,KE,ST)
750 IF ST=0 THEN 740
760 CALL CLEAR
770 PRINT "SIE MUESSEN NUN DIE GOLD-    B
ARREN AUS DER SCHATZTRUHE  HOLEN, UND DA
NN ZUM SCHIFF   ZURUECKSCHWIM-"
780 PRINT "MEN, UM DAS GOLD ABZULEGEN   U
ND SAUERSTOFF ZU TANKEN.   SIE KOENNEN N
ICHT MEHR ALS   5 GOLDBARREN "
790 PRINT "AUF EINMAL NEHMEN.            U
ND VORSICHT! BEI MEHR ALS  3 GOLDBARREN
VERBRAUCHEN SIEMEHR LUFT."
800 PRINT "GOLD NIMMT MAN, INDEM MAN    U
EBER DER SCHATZTRUHE DIE    TASTE NACH UN
TEN DRUECKT."
810 PRINT
820 PRINT "STEUERTASTEN:E,X,S,D
     (IN PFEILRICHTUNG) "
830 PRINT
840 PRINT "AUFGEBEN:TASTE Q"
850 PRINT
860 PRINT "  SPIELANFANG MIT
  SPACE-TASTE"
870 CALL KEY(0,KE,ST)
880 IF ST=0 THEN 870
890 CALL CLEAR
900 CALL SCREEN(5)
910 RESTORE
920 CALL CHAR(36,"4BFFFBBED76DFBFC")
930 CALL CHAR(33,"FF7F3F1F0F070301")
940 CALL CHAR(34,"80C0E0F0F8FCFEFF")
950 CALL CHAR(35,"0103070F1F3F7FFF")
960 CALL CHAR(37,"0000011FFF3F0301")
970 CALL CHAR(38,"008081F3FFFB0180")
980 CALL CHAR(39,"FFFFFFFFFFFFFFFF")
990 CALL CHAR(40,"FF7F3F1F0F070301")
1000 CALL CHAR(41,"FFFEFCF8F0E0C080")
1010 CALL CHAR(43,"0101010101010101")
1020 CALL CHAR(60,"4BFFFBBED76DFBFC")
1030 CALL CHAR(152,"FFFFFBFFFFBFFDFF")
1040 CALL CHAR(88,"FFFFDFBFFFDFFFFF")
1050 CALL CHAR(96,"00C1390707784000")
1060 CALL CHAR(97,"0CF2F7FFF7331008")
1070 CALL CHAR(59,"000008143A565EF7")
1080 CALL CHAR(104,"000000F0FFF82710")
1090 CALL CHAR(105,"000000073FCF8442")
1100 CALL CHAR(112,"E0380E03FFD5ABFF")
1110 CALL CHAR(120,"FFFFFFFFFFFFFFFF")
1120 CALL CHAR(121,"0103070F1F3F7FFF")
1130 CALL CHAR(128,"FFFFFFFFFFFFFFFF")
1140 CALL CHAR(136,"0000000000000000")
1150 CALL CHAR(144,"20100B3C4E9D2A4A")
1160 CALL CHAR(145,"4991101010303030")
1170 CALL COLOR(1,8,5)
1180 CALL COLOR(2,8,6)
1190 CALL COLOR(3,8,5)
1200 CALL COLOR(4,8,5)
1210 FOR I=5 TO 7
1220 CALL COLOR(I,8,5)
1230 NEXT I
1240 CALL COLOR(8,4,7)
1250 CALL COLOR(9,2,5)
1260 CALL COLOR(10,4,5)
1270 CALL COLOR(11,11,5)
1280 CALL COLOR(12,16,6)
1290 CALL COLOR(13,6,5)
1300 CALL COLOR(14,5,5)
1310 CALL COLOR(15,2,6)
1320 CALL CLEAR
1330 CALL COLOR(16,15,7)
1340 FOR Z=1 TO 48
1350 READ X,Y,B,C
1360 CALL HCHAR(X,Y,B,C)
1370 NEXT Z
1380 DATA 24,5,36,10,24,24,36,1,23,12,36
,2,23,15,36,18,22,16,36,11,22,30,36,3
1390 DATA 21,16,36,2,21,20,36,1,21,28,36
,2,20,15,36,2,20,23,36,2,20,28,36,2
1400 DATA 19,14,36,1,19,18,36,1,19,22,36
,1,19,25,36,1,18,17,36,1,18,24,36,1
1410 DATA 17,21,36,1,17,23,36,2,17,27,36
,1,17,31,36,1,16,22,36,2,16,30,36,2
1420 DATA 15,18,36,15,14,17,36,6,14,24,3
6,9,13,26,36,7,12,31,36,2,22,29,33,1
1430 DATA 21,27,33,1,18,16,33,1,17,20,33
,1,19,19,34,1,19,23,34,1,19,29,34,1
1440 DATA 21,19,35,1,21,31,112,1,1,1,128
,256,8,15,39,3,8,14,40,1,8,18,41,1,8,1,1
52,8
1450 DATA 7,1,88,5,6,1,88,3,7,16,120,2,7
,15,121,1,18,28,36,1
1460 FOR Z=1 TO 10
1470 READ X,Y,B,C
1480 CALL VCHAR(X,Y,B,C)
1490 NEXT Z
1500 DATA 12,32,36,13,5,16,43,2,9,1,60,1
0,9,2,60,9,9,3,60,8,9,4,60,7,9,5,60,6
1510 DATA 9,6,60,5,9,7,60,3,9,8,60,1
1520 CALL VCHAR(19,1,60,6)
1530 CALL VCHAR(18,2,60,7)
1540 CALL VCHAR(21,3,60,4)
1550 CALL VCHAR(23,4,60,2)
1560 CALL HCHAR(4,2,144)
1570 CALL HCHAR(5,2,145)
1580 CALL HCHAR(5,4,144)
1590 CALL HCHAR(6,4,145)
1600 RANDOMIZE
1610 H=0
1620 K=0
1630 R=0
1640 GOLD=0
1650 GG=0
1660 LUFT=131
1670 X=9
1680 Y=17
1690 Z=16
1700 XOLD=X
1710 ZOLD=Z
1720 CALL SOUND(600,391,0,329,0,261,0)
1730 IF R>0 THEN 3050
1740 R=0
1750 CALL HCHAR(XOLD,ZOLD,136,2)
1760 CALL SOUND(400,-4,0)
1770 CALL HCHAR(X,Y,97)
1780 CALL HCHAR(X,Z,96)
1790 S=1
1800 IF GOLD>3 THEN 1820
1810 GOTO 1830
1820 S=2
1830 LUFT=LUFT-S
1840 IF LUFT>10 THEN 1870
1850 CALL SOUND(300,391,0)
1860 CALL SOUND(300,329,0)
1870 DRUCK$="LUFT="&STR$(LUFT)&" "
1880 HTAB=15
1890 GOSUB 2990
1900 IF LUFT<=0 THEN 3430
1910 IF X=9 THEN 3340
1920 XOLD=X
1930 YOLD=Y
1940 ZOLD=Z
1950 IF GOLD>5 THEN 3410
1960 DRUCK$="GOLD="&STR$(GOLD)
```

Scan: S. Höltgen


```
1970 HTAB=25
1980 IF GG>=20 THEN 3510
1990 GOSUB 2990
2000 CALL KEY(0,A,B)
2010 IF A=68 THEN 3240
2020 IF A=83 THEN 3270
2030 IF A=69 THEN 3300
2040 IF A=88 THEN 3320
2050 IF A=81 THEN 3650
2060 IF X=8 THEN 2940
2070 IF Y=33 THEN 2960
2080 CALL GCHAR(X,Y,C)
2090 GOSUB 2890
2100 CALL GCHAR(X,Z,C)
2110 GOSUB 2890
2120 IF H=1 THEN 2240
2130 D=INT(20*RND)
2140 IF D=0 THEN 2160
2150 GOTO 2340
2160 CALL SOUND(500,130,0,261,0)
2170 DRUCK$="ACHTUNG  "
2180 HTAB=15
2190 GOSUB 2990
2200 H=1
2210 P=10
2220 Q=9
2230 QOLD=Q
2240 CALL HCHAR(9,QOLD,136,2)
2250 CALL HCHAR(9,P,104)
2260 CALL HCHAR(9,Q,105)
2270 QOLD=Q
2280 P=P+1
2290 Q=Q+1
2300 IF P=33 THEN 2320
2310 GOTO 2340
2320 CALL HCHAR(9,31,136,2)
2330 H=0
2340 IF K=1 THEN 2480
2350 E=INT(20*RND)
2360 IF E=0 THEN 2380
2370 GOTO 2620
2380 CALL SOUND(500,130,0,261,0)
2390 DRUCK$="ACHTUNG  "
2400 HTAB=15
2410 GOSUB 2990
2420 K=1
2430 U=11
2440 V=31
2450 W=32
2460 UOLD=U
2470 VOLD=V
2480 CALL HCHAR(UOLD,VOLD,136,2)
2490 CALL HCHAR(U,V,37)
2500 CALL HCHAR(U,W,38)
2510 UOLD=U
2520 VOLD=V
2530 IF U=23 THEN 2600
2540 IF V=10 THEN 2580
2550 V=V-1
2560 W=W-1
2570 GOTO 2620
2580 U=U+1
2590 GOTO 2620
2600 CALL HCHAR(23,10,136,2)
2610 K=0
2620 F=INT(3000*RND)
2630 IF F=2 THEN 2650
2640 GOTO 2730
2650 CALL SOUND(1000,-7,0)
2660 CALL HCHAR(21,23,59)
2670 DRUCK$="LAWINE   "
2680 HTAB=15
2690 GOSUB 2990
2700 FOR I=1 TO 400
2710 NEXT I
2720 GOTO 3590
2730 CALL GCHAR(X,Y,C)
2740 GOSUB 2780
2750 CALL GCHAR(X,Z,C)
2760 GOSUB 2780
2770 GOTO 1730
2780 FOR I=33 TO 36
2790 IF C=I THEN 3050
2800 NEXT I
2810 IF C=37 THEN 3430
2820 IF C=38 THEN 3430
2830 IF C=104 THEN 3430
2840 IF C=105 THEN 3430
2850 IF C=59 THEN 3050
2860 IF C=60 THEN 3050
2870 IF C=112 THEN 3210
2880 RETURN
2890 IF C=37 THEN 3430
2900 IF C=38 THEN 3430
2910 IF C=104 THEN 3430
2920 IF C=105 THEN 3430
2930 RETURN
2940 X=X+1
2950 GOTO 2070
2960 Z=Z-1
2970 Y=Y-1
2980 GOTO 2080
2990 FOR DRI=1 TO LEN(DRUCK$)
3000 CODE=ASC(SEG$(DRUCK$,DRI,1))
3010 CALL HCHAR(24,HTAB,CODE)
3020 HTAB=HTAB+1
3030 NEXT DRI
3040 RETURN
3050 X=XOLD
3060 Y=YOLD
3070 Z=ZOLD
3080 CALL SOUND(300,783,0,391,0)
3090 DRUCK$="BEWUSTLOS"
3100 HTAB=15
3110 GOSUB 2990
3120 DRUCK$="          "
3130 HTAB=15
3140 GOSUB 2990
3150 C=0
3160 IF R>0 THEN 3190
3170 R=4
3180 GOTO 1750
3190 R=R-1
3200 GOTO 1750
3210 GOLD=GOLD+1
3220 X=XOLD
3230 GOTO 1730
3240 Z=Z+1
3250 Y=Y+1
3260 GOTO 2060
3270 Y=Y-1
3280 Z=Z-1
3290 GOTO 2060
3300 X=X-1
3310 GOTO 2060
3320 X=X+1
3330 GOTO 2060
3340 IF Y>17 THEN 1920
3350 IF Y<15 THEN 1920
3360 GG=GG+GOLD
3370 CALL SOUND(150,783,0)
3380 GOLD=0
3390 LUFT=131
3400 GOTO 1920
3410 GOLD=GOLD-1
3420 GOTO 1960
3430 GOSUB 3670
3440 PRINT "DIE GEFAHREN IM MEER
SIND GROSS."
3450 PRINT "SIE SIND IHNEN ERLEGEN."
3460 IF GG=0 THEN 3490
3470 PRINT "DIE";GG;"BARREN GOLD WERDEN"
3480 PRINT "UNTER IHREN VERWANDTEN
VERTEILT."
3490 PRINT :::
3500 GOTO 3770
3510 GOSUB 3670
3520 PRINT "BRAVO. SIE HABEN ALLE
ABENTEUER BESTANDEN."
3530 PRINT "UND HABEN DIE GANZE SCHATZ-
TRUHE AUSGERAEUMT."
3540 PRINT "JETZT KOENNEN SIE MIT DEN"
3550 PRINT GG;"BARREN GOLD EIN GERUH-"
3560 PRINT "SAMES LEBEN FUEHREN."
3570 PRINT :::
3580 GOTO 3770
3590 GOSUB 3670
3600 GG=GOLD+GG
3610 PRINT "SO EIN PECH.
EIN STEINSCHLAG HAT DIE      GROTTE VERSC
HUETTET."
3620 PRINT "GOLDBARREN=";GG
3630 PRINT :::
3640 GOTO 3770
3650 GOSUB 3670
3660 GOTO 3730
3670 CALL CLEAR
3680 CALL SCREEN(5)
3690 FOR I=1 TO 12
3700 CALL COLOR(I,16,5)
3710 NEXT I
3720 RETURN
3730 PRINT "WARUM HABEN SIE DENN AUFGE-
HOERT?                        ES WAREN 20
BARREN GOLD IN"
3740 PRINT "DER SCHATZTRUHE UND SIE
HABEN NUR"
3750 PRINT GG;"BARREN GOLD GEBORGEN."
3760 PRINT :::
3770 FOR N=500 TO 800 STEP 50
3780 CALL SOUND(100,N,0)
3790 NEXT N
3800 FOR N=800 TO 400 STEP -50
3810 CALL SOUND(100,N,0)
3820 NEXT N
3830 INPUT " NOCH EIN SPIEL J/N ?":M$
3840 IF M$="J" THEN 890
3850 PRINT
3860 PRINT "    SCHADE..."
3870 FOR I=1 TO 300
3880 NEXT I
3890 CALL CLEAR
3900 END
```

Scan: S. Höltgen

```
5 : LET F(Q)=39
9860 IF MS=14 THEN LET PA1=0: LE
T PASCH=0: LET F(Q)=100
9865 PAUSE 100
9870 INK 0: GO TO 8020
9900 INK 1: CLS
9901 LET PASCH=0: LET PA1=0
9910 PLOT 30,0: DRAW 0,100: DRAW
 0,-50,-PI: CIRCLE 50,30,20: PLO
T 70,100: DRAW 0,-100: DRAW 50,0
: PLOT 80,100: DRAW 0,-80: PLOT
75,20: DRAW 10,0: PLOT 75,100: D
RAW 10,0: PLOT 100,100: DRAW 60,
0: DRAW -60,-90: DRAW 60,0
9920 PLOT 230,100: DRAW -60,0: D
RAW 0,-100: DRAW 60,0: PLOT 170,
80: DRAW 40,0: PLOT 235,90: DRAW
 10,0: PLOT 240,90: DRAW 0,-88:
PLOT 235,2: DRAW 10,0
9930 PRINT AT 0,4;"GEHE IN DAS G
EFAENGNIS !";AT 4,0;P$: LET P$="
"
9935 PAUSE 111
9940 INK 0: LET F(Q)=100: RETURN
9950 REM FP
9955 CLS : INK 3
9960 INK 2: CIRCLE 40,60,20: CIR
CLE 140,60,20: PLOT 10,60: DRAW
170,0: DRAW -30,40,2: DRAW -20,2
0: DRAW -80,0: DRAW -40,-60,1
9970 PLOT 150,100: DRAW -10,-5:
DRAW -118,0: PLOT 130,95: DRAW 0
,25: PLOT 50,95: DRAW 0,25
9975 FOR N=90 TO 95: PLOT N,95:
DRAW 0,25: NEXT N: DRAW 0,-60: P
LOT 130,95: DRAW 0,-17
9980 PRINT AT 0,10;"FREI PARKEN
!";AT 3,0;"DU ERHAELST M ";FP;".
- !"
9985 LET G(Q)=G(Q)+FP: INK 0
9986 LET FP=0
9987 PAUSE 111
9990 RETURN
9999 SAVE "MONOPOL.MS" LINE 1
```



Das Programm stellt Aufgaben aus zwei Zahlen, die je nach Rechenart mit +, −, * oder : verbunden sind, und kontrolliert das Ergebnis. Bei richtig eingetipptem Ergebnis klettert ein Männchen die nächste Sprosse einer Leiter hinauf. Nach sieben erfolgreich gelösten Aufgaben schiebt das Männchen die Buchstaben „BRAVO" auf den Bildschirm. Ist eine Aufgabe beim drittenmal nicht gelöst, wird das Ergebnis angezeigt. Danach muß die Aufgabe erneut gelöst werden.

Am Beispiel einer Additionsaufgabe soll verdeutlicht werden, in welchen Formen die Aufgaben gestellt werden:

Rechentrainer

Zum spielerischen Erlernen der vier Grundrechenarten: ein gründlich durchdachtes, abwechslungsreiches, im Programmierstil vorbildliches Programm für den TI-99/4A mit Extended-BASIC

```
 7 + 12 =  ?
 7 +  ? = 19
 ? + 12 = 19
19 =  7 +  ? (Ergebnis
             zerlegen)
```

Bei der Addition, Subtraktion und Multiplikation kann außerdem schriftliches Rechnen simuliert werden. Die Zahlen werden dann untereinander auf dem Bildschirm ausgegeben, und es wird immer das Ergebnis, bei der Multiplikation auch Zwischenergebnisse, verlangt. Es kann gewählt werden:

a) der Zahlenbereich (wird, wenn nötig, automatisch vergrößert)

b) die Rechenart

c) Kopfrechnen (oder schriftliches Rechnen)

d) Aufgabenstellung in der

Scan: S. Höltgen

Form „Ergebnis zerlegen"

e) Aufgaben mit relativen Zahlen (nur bei Kopfrechnen)

f) ob bei der Addition Zehner oder Hunderter überschritten werden dürfen

Bei Punktrechnung noch zusätzlich:

g) ein bestimmtes Einmaleins

h) Aufgaben nur aus dem kleinen Einmaleins

i) Aufgaben nur aus dem großen Einmaleins

j) beliebige Aufgaben innerhalb des Zahlenbereichs

k) Teilen mit Rest

Nach jeder Aufgabe kann man die Bedingungen neu wählen.

Sprosse für Sprosse

Nach dem Titelbild aus Ziffern und Rechenoperationszeichen werden einzeln die Bedingungen abgefragt, denen die Aufgabenstellung genügen soll. Der Schirm wird gelöscht, links im Bild erscheint eine Leiter, und das Männchen erklimmt die Sprosse 0. Im rechten Bildteil erscheint eine Rechenaufgabe.

An der Stelle, wo der Cursor abwechselnd mit dem Fragezeichen blinkt, soll die gesuchte Zahl eingetippt werden. Nach jedem Tastendruck springt der Cursor wie gewohnt nach rechts weiter, wenn man im Kopf rechnet. Beim schriftlichen Rechnen springt der Cursor nach links weiter, so daß man erst die Einer, dann die Zehner und so weiter errechnen kann. Mit ENTER beendet man die Eingabe. Vorher kann man mit FCTN S und FCTN D falsche Zahlen korrigieren.

Der Computer prüft, ob die eingetippte Zahl richtig ist. Wenn nicht, ertönt ein Signalton mit der Schrift: „Nicht richtig, bitte noch einmal." Hat man bei negativen Zahlen die Klammern vergessen, wird man darauf hingewiesen. Mit ENTER läuft das Programm weiter, dieselbe Aufgabe erscheint erneut.

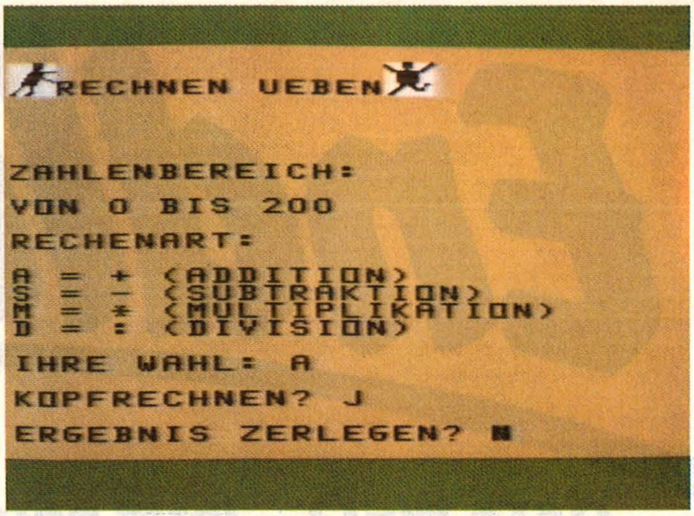
Bei richtiger Lösung klettert das Männchen mit Tönen eine Sprosse weiter nach oben. Nach der 7. Stufe schiebt es dann die Buchstaben BRAVO auf den Bildschirm.

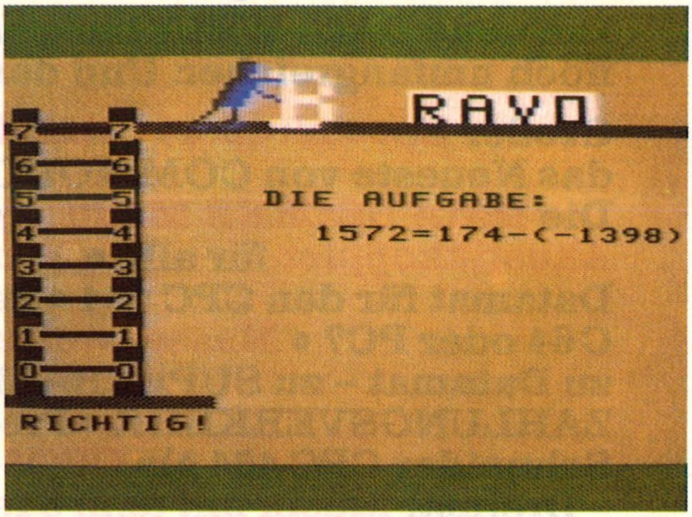
Beantwortet man die Frage: „Möchtest Du weitere Aufgaben?" mit ja, erscheint wieder die Bedingungsliste, andernfalls stoppt das Programm.

Beim Titelvorspann werden einige Variable in anderer Funktion benutzt, als in der Variablenliste angegeben. In den Unterprogrammen wurden die Variablennamen soweit wie möglich beibehalten. Wo nur ein Wert eines Arrays gebraucht wird, verwandelt sich die indizierte Variable des Hauptprogramms in eine einfache Variable des Unterprogramms. Im Listing bedeutet ! ein REM. Alle Zeilen mit ! können weggelassen werden.

Bernhard Decker

```
100 OPTION BASE 1
110 DIM Z(3),Z$(4),X$(3)
120 !TITEL
130 RANDOMIZE :: CALL MAGNIFY(2):: CALL
CLEAR :: CALL SCREEN(2)
140 A$="1+2=347*5-9:608"
150 J=1 :: FOR I=1 TO 15
160 T=ASC(SEG$(A$,J,1))
170 K=INT(RND*192)+1 :: S=INT(RND*256)+1
 :: D=INT(RND*40)-19 :: L=INT(RND*40)-19
180 C=INT(RND*15)+2 :: IF C<3 THEN 180
190 CALL SOUND(-1000,100+K+S,6):: CALL S
PRITE(#I,T,C,70,120,D,L):: J=J+1
200 NEXT I
210 FOR I=2 TO 8 :: CALL COLOR(I,2,11)::
 NEXT I
220 J=15 :: C=2 :: GOSUB 1840
230 CALL COLOR(9,2,11,11,2,16):: CALL CH
AR(96,"00FFFFFFFFFFFFFF0000FFFFFF000000A
A55AA55AA550000AA55FFFFFF550000")
240 CALL CHAR(100,"0101010101010101180808
080808080"):: GOSUB 1840
250 DATA 070707073F7F5F5F5F5F1F050505050
500000000E0F0D0D0D0D0C0000000000
260 DATA 474747427F1F1F1F1F1F05050501010
110101010F0C0C0C0C0C000000000000
270 DATA 0383C3613F0F0F0F0F0F07060C18306
08183860CF8E0E0E0E0E0C6C468380000
280 DATA 00000000001010303070704081021б
370707060FFF0F0E8E6C1C08080808000
290 DATA 0000000000000000000000000000010
```





Variable	Bedeutung
A	Stelle der gesuchten Zahl
A$	Zwischenspeicher für Stringvariable
B	Zahlenbereich
B$	Zwischenspeicher
C	Das gewählte Einmaleins
D	Zeilenberechnung für Lösungsangabe
F	Zähler für Fehler
F1	Zähler für richtige Aufgaben
G	Flag; für Schwierigkeitsgrade
G1	Flag 1 = Teilen ohne, 2 = Teilen mit Rest
G2	Flag 1 = Kopfrechnen, 2 = schriftlich rechnen
G3	Flag, wenn 2, ist Ergebnis die gesuchte Zahl
G4	Flag 1 = Aufgabenstellung normal, 2 = Ergebnis zerlegen
H$	Zwischenspeicher bei relativen Zahlen
I, J	Schleifenzähler und Hilfsspeicher
K	Variable für ASCII in Key-Routine
L	Möglicher Platzbedarf für gesuchte Zahl
L1	Platzlänge der Aufgabe
M	Rechenart
P	Spalte für Eingabe der gesuchten Zahl
R	Rest
RZ	Flag = 2 bei Rechnen mit relativen Zahlen
S	Status in Key-Routine
T	10 = Zehner, 100 = Hunderter nicht überschreiten
V$	Speicher für Rechenoperationszeichen
X$()	Positionen in Aufgabenstellung, zum Beispiel: X$(1)&V$&X$(2)&"="&X$(3)
Z()	Zahlen der Aufgabe, zum Beispiel: Z(1)+Z(2)=Z(3)
Z$()	String von Z()

Variablenliste

Scan: S. Höltgen

Zeilen	Kommentar
10– 90	Impressum
100– 360	Initialisierung und Titel
230– 240	Charakterdefinition für Leiterelemente
250– 290	DATA für Charakter 104– 123: Männchen
300– 340	DATA für Charakter 124–143: BRAVO
370– 790	Abfrage der Bedingungen der Aufgabenstellung
800–1530	Programmteil Aufgaben stellen
820	Verzweigung nach Rechenarten
840– 900	Zahlen der Aufgabe als String formatieren
920–1320	Aufgabe auf Bildschirm schreiben
940– 980	Bei Kopfrechnen
1070–1150	Bei schriftlicher Strichrechnung
1170–1320	Bei schriftlicher Multiplikation
990–1050	Bewertung
1340–1530	Wenn 7. Aufgabe gelöst, dann
1390–1480	BRAVO auf Schirm schieben
1550	Zeitverzögerung
1570–1590	Löscht Schreibfeld
1610–1620	Hilfsroutine bei Bedingungsabfrage
1650–1780	Leiter und Klettern
1650–1680	Malt Leiter
1710–1780	Männchen klettert
1800–1820	Löscht bestimmte Zeilen
1840–1870	Routine für Schrift „RECHEN-TRAINER“
1890–1960	Wahl der Zahlen bei Addition
1980–2050	Wahl der Zahlen bei Subtraktion
2070–2260	Wahl der Zahlen bei Multiplikation
2280–2470	Wahl der Zahlen bei Division
2480–2880	Unterprogramme
2480–2510	Wechselt per Zufall Vorzeichen
2520–2540	Setzt Klammern um negative Zahlen
2550–2720	Editor bei schriftlichem Rechnen
2730–2840	Prüft Lösung auf Richtigkeit
2850–2880	Programmstop bis Tastendruck

Programmstruktur

```
1383838107F787CFAF1F0F090888C0404
300 DATA 00003F3F0C0C0C0C0F0F0C0C0C0C3F3
F0000C0C030303030C0C030303030C0C0
310 DATA 00003F3F303030303F3F33333030303
00000C0C030303030C0C00000C0C03030
320 DATA 00000F0F303030303F3F30303030303
00000C0C030303030F0F030303030303030
330 DATA 0000303030303030300C0C0C0C0303030
30000303030303030C0C0C0C000000000
340 DATA 00003F3F30303030303030303030303F3
F0000F0F030303030303030303030F0F0
350 GOSUB 1840 :: FOR I=1 TO 5 :: READ A
$ :: CALL CHAR(100+4*I,A$):: NEXT I :: G
OSUB 1840 :: FOR I=6 TO 10 :: READ A$ ::
 CALL CHAR(100+4*I,A$):: NEXT I
360 FOR I=0 TO 8 :: CALL COLOR(I,2,11)::
 NEXT I :: CALL COLOR(12,2,16,13,2,16,14
,2,16)
370 CALL CLEAR :: CALL DELSPRITE(ALL)::
CALL SCREEN(13):: CALL MAGNIFY(4)
380 PRINT "tv"," pr":"uwRECHNEN UEBENqs"
: : : : :"ZAHLENBEREICH:": :
390 PRINT "VON 0 BIS 100"
400 ACCEPT AT(23,11)SIZE(-5)VALIDATE(NUM
ERIC):A$
410 IF LEN(A$)=0 THEN 400 ELSE B=MAX(VAL
(A$)+1,3)
420 PRINT :"RECHENART:": :"A = + (ADDITI
ON)":"S = - (SUBTRAKTION)":"M = * (MULTI
PLIKATION)":"D = : (DIVISION)"
430 PRINT :"IHRE WAHL: A"
440 ACCEPT AT(23,12)VALIDATE("ASMD")SIZE
(-1):A$
450 M=POS("ASMD",A$,1):: T,G=0 :: G1,G2,
G3,G4,RZ=1
460 IF M<4 THEN PRINT :"KOPFRECHNEN? J"
ELSE 490
470 I=14 :: GOSUB 1610
480 IF A$="N" THEN G2=2 :: GOTO 560
490 PRINT :"ERGEBNIS ZERLEGEN? N"
500 I=20 :: GOSUB 1610
510 IF A$="J" THEN G4=2 :: G3=2
520 IF G2=2 THEN 560
530 PRINT :"RELATIVE ZAHLEN? N"
540 I=18 :: GOSUB 1610
550 IF A$="J" THEN RZ=2 !=FLAG RELATIVE
ZAHL
560 L=LEN(STR$(B)):: IF RZ=1 THEN L=L+1
!L=MOEGL PLATZBEDARF
570 IF B>101 OR M>2 THEN 620
580 PRINT :"ZEHNER UEBERSCHREITEN? J"
590 I=24 :: GOSUB 1610
600 IF A$="N" THEN G3=2 :: G=4 :: T=10 !
ZAHLENFEINBEREICH
610 GOTO 800
620 IF B>1001 OR M>2 THEN 650 ELSE PRINT
 :"HUNDERTER UEBERSCHREITEN? N"
630 I=27 :: GOSUB 1610
640 IF A$="N" THEN G3=2 :: G=4 :: T=100
!ZAHLENFEINBEREICH
650 IF M<3 THEN 800
660 IF G2=2 THEN 790
670 PRINT :"EIN BESTIMMTES 1*1 ? J" :: I
=22 :: GOSUB 1610
680 IF A$="N" THEN 720
690 PRINT :"UND ZWAR DAS 1*2"
700 ACCEPT AT(23,16)VALIDATE(DIGIT)SIZE(
-2):C
710 G=4 :: G3=2 :: GOTO 750
720 PRINT :"KLEINES 1*1 ? J" :: I=15 ::
GOSUB 1610
730 IF A$="J" THEN G=1 :: GOTO 800 ELSE
PRINT :"GROSSES 1*1 ? J" :: I=15 :: GOSU
B 1610
740 IF A$="J" THEN G=2 ELSE G=3
750 IF M<4 OR G4=2 THEN 800
760 PRINT :"TEILEN MIT REST? N" :: I=18
:: GOSUB 1610
770 IF A$="J" THEN G1=2
780 GOTO 800
790 G=3 :: IF B<401 THEN B=401 !SCHRFTL
MULTIPLIZIEREN
800 CALL CLEAR
810 GOSUB 1650
820 F=0 :: ON M GOSUB 1890,1980,2070,228
0
830 ! ZAHLEN IN STRING FORMATIEREN
840 IF G4=1 THEN 850 ELSE I=Z(1):: Z(1)=
Z(3):: Z(3)=Z(2):: Z(2)=I
850 Z$(1)=STR$(Z(1)):: Z$(2)=STR$(Z(2)):
: IF G1=1 THEN Z$(3)=STR$(Z(3))
860 IF G4=1 THEN CALL KLAMMER(Z(2),Z$(2)
)ELSE CALL KLAMMER(Z(3),Z$(3))
870 IF G2=2 OR G3=2 THEN A=3 ELSE A=INT(
RND*3)+1
880 FOR I=1 TO 3 :: IF I<>A THEN X$(I)=Z
$(I):: GOTO 900 ELSE X$(I)="?"
890 IF G2=1 THEN X$(I)="?"&RPT$(" ",MAX(
L,LEN(Z$(I)))-1)ELSE X$(I)="?"
900 NEXT I
910 ! AUFGABE SCHREIBEN
920 GOSUB 1570 :: DISPLAY AT(9,11):"DIE
AUFGABE:"
930 ON G2 GOTO 940,1070
940 IF G4=1 THEN A$=X$(1)&V$&X$(2)&"="&X
$(3)ELSE A$=X$(1)&"="&X$(2)&V$&X$(3)
950 GOSUB 1580 :: ON G2 GOTO 960,1100
960 DISPLAY AT(11,29-LEN(A$)):A$
970 P=28-LEN(A$)+POS(A$,X$(A),1):: L1=MA
X(L,LEN(X$(A))):: D=0
980 ACCEPT AT(11,P)SIZE(-L1)VALIDATE(NUM
ERIC,"REST ()"):Z$(4)
990 CALL CHECK(Z$(4),Z$(A),F,G2,D,L1,A$,
P)
1000 IF F>0 THEN 950
1010 F1=F1+1 :: DISPLAY AT(22,1):"RICHTI
G!"
1020 CALL COLOR(#1,5):: GOSUB 1710
1030 IF F1=7 THEN 1340
1040 DISPLAY AT(23,11):"NEUE AUFGABE? J"
 :: I=25 :: GOSUB 1610
1050 IF A$="J" THEN 820 ELSE 370
1060 !SCHRIFTLICH RECHNEN
1070 ON M GOTO 1090,1090,1170
1080 !ADDITION & SUBTRAKTION
1090 A$=Z$(A):: L1=LEN(A$):: D=4 :: P=29
-LEN(Z$(A))
1100 DISPLAY AT(11,29-LEN(X$(1))):X$(1)
1110 DISPLAY AT(12,23):V$;TAB(29-LEN(X$(
2)));X$(2)
1120 DISPLAY AT(13,23):"------" :: DISPL
AY AT(14,23):"="
1130 DISPLAY AT(14,29-LEN(X$(3))):X$(3)
1140 CALL EDIT(A+1,30,Z$(4))
```

Scan: S. Höltgen


```
1150 GOTO 990
1160 !MULTIPLIKATION
1170 B$=X$(1)&V$&X$(2):: L1=LEN(B$):: D=
LEN(Z$(2))+3
1180 DISPLAY AT(11,29-L1)SIZE(L1):B$ ::
DISPLAY AT(12,29-L1)SIZE(L1):RPT$("-",L1
)
1190 DISPLAY AT(13+LEN(X$(2)),29-L1)SIZE
(L1):RPT$("-",L1)
1200 FOR I=1 TO LEN(X$(2))
1210 J=VAL(SEG$(X$(2),I,1))*Z(1):: A$,Z$
(A)=STR$(J):: P=30-LEN(X$(2))+I
1220 GOSUB 1800
1230 CALL EDIT(2+I,P,Z$(4))
1240 P=29-LEN(A$):: CALL CHECK(Z$(4),A$,
F,G2,D,L1,A$,P)
1250 IF F>0 THEN 1210
1260 NEXT I
1270 DISPLAY AT(13+I,28-L1):"=" :: GOSUB
 1810
1280 A$,Z$(A)=STR$(Z(1)*Z(2))
1290 CALL EDIT(3+I,30,Z$(4))
1300 P=29-LEN(A$):: CALL CHECK(Z$(4),A$,
F,G2,D,L1,A$,P)
1310 IF F>0 THEN 1270
1320 GOTO 1010
1330 !7 AUFGABEN GELOEST
1340 GOSUB 1550 :: FOR L=1 TO 3
1350 CALL SOUND(-400,1046,2,783,5,659,5)
:: GOSUB 1550
1360 NEXT L
1370 CALL DELSPRITE(ALL)
1380 FOR I=1 TO 5
1390 A=-4 :: L=116 :: J=32 :: CALL SPRIT
E(#I,144-4*I,16,4,J,#6,116,5,4,J-24)
1400 CALL COINC(#I,4,216-16*I,10,C)
1410 IF C=-1 THEN 1440
1420 J=J+8 :: CALL SOUND(-50,-7,10)
1430 A=-A :: L=L+A :: CALL PATTERN(#6,L)
:: CALL LOCATE(#I,4,J,#6,4,J-24):: GOTO
1400
1440 CALL SOUND(200,-6,2):: CALL DELSPRI
TE(#6)
1450 CALL HCHAR(3,27-2*I,144-4*I):: CALL
 HCHAR(4,27-2*I,145-4*I)
1460 CALL HCHAR(3,28-2*I,146-4*I):: CALL
 HCHAR(4,28-2*I,147-4*I)
1470 CALL DELSPRITE(#I):: NEXT I
1480 CALL DELSPRITE(#6)
1490 CALL HCHAR(3,15,116):: CALL HCHAR(4
,15,117):: CALL HCHAR(3,16,118):: CALL H
CHAR(4,16,119):: GOSUB 1570
1500 CALL MAGNIFY(4):: CALL SPRITE(#1,11
2,16,65,68)
1510 DISPLAY AT(10,10):"MOECHTEST DU" ::
 DISPLAY AT(11,10):"WEITERE AUFGABEN? J"
1520 ACCEPT AT(11,28)VALIDATE("JN")SIZE(
-1):A$
1530 IF A$="J" THEN F1=0 :: GOTO 370 ELS
E STOP
1540 !VERZOEGERN
1550 FOR J=1 TO 20 :: NEXT J :: RETURN
1560 !SCHREIBFELD LOESCHEN
1570 FOR I=9 TO 12 :: CALL HCHAR(I,11,32
,21):: NEXT I
1580 FOR I=13 TO 21 :: CALL HCHAR(I,11,3
2,21):: NEXT I
1590 CALL HCHAR(22,1,32,96):: RETURN
1600 !INPUT
1610 ACCEPT AT(23,I)VALIDATE("JN")SIZE(-
1):A$
1620 IF LEN(A$)=0 THEN 1610
1630 RETURN
1640 !LEITER
1650 CALL HCHAR(5,1,98,32):: CALL VCHAR(
6,2,100,15):: CALL VCHAR(4,3,96,17)
1660 CALL VCHAR(4,7,96,17):: CALL VCHAR(
6,8,101,15):: CALL HCHAR(21,1,98,10)
1670 FOR I=0 TO 7 :: CALL HCHAR(19-I*2,4
,97,3):: IF I>F1 THEN 1680 ELSE CALL HCH
AR(19-I*2,3,48+I):: CALL HCHAR(19-I*2,7,
48+I)
1680 NEXT I :: CALL HCHAR(5,4,99,3)
1690 IF F1>0 THEN 1710
1700 CALL SPRITE(#1,104,16,158,24):: GOS
UB 1550
1710 FOR I=1 TO 3 :: CALL SOUND(-100,209
2,10):: CALL PATTERN(#1,112):: CALL COLO
R(1,1,9,9,16,9):: GOSUB 1550
1720 CALL SOUND(-100,2092,10):: CALL PAT
TERN(#1,104):: CALL COLOR(1,2,11,9,2,11)
:: GOSUB 1550 :: NEXT I
1730 CALL SOUND(100,2092,6):: CALL LOCAT
E(#1,147-16*F1,24):: GOSUB 1550
1740 FOR I=1 TO 2 :: CALL SOUND(-400,329
,6):: CALL SPRITE(#1,108,7,143-16*F1,24)
:: GOSUB 1550
1750 IF I=2 THEN 1770
1760 CALL SOUND(-400,261,6):: CALL SPRIT
E(#1,104,7,150-16*F1,24):: GOSUB 1550 ::
 NEXT I
1770 CALL SOUND(600,391,5,329,6,261,6)::
 CALL SPRITE(#1,104,16,146-16*(F1+1),24)
:: GOSUB 1550
1780 CALL HCHAR(19-2*F1,3,48+F1):: CALL
HCHAR(19-2*F1,7,48+F1):: CALL SOUND(700,
783,8,658,9,523,10):: RETURN
1790 !ZEILEN LOESCHEN
1800 CALL HCHAR(12+I,23,32,10)
1810 CALL HCHAR(13+D,13,32,19):: FOR J=1
9 TO 22 :: CALL HCHAR(J,11,32,21):: NEXT
 J :: CALL HCHAR(23,1,32,64)
1820 RETURN
1830 !TITELHILFSROUTINE
1840 FOR I=1 TO 19 :: DISPLAY AT(I+2,1):
"" :: NEXT I
1850 A$="RECHEN-TRAINER"
1860 C=-C :: CALL SOUND(-4000,1367-C*100
,5,974-C*100,5,787-C*100,5):: FOR I=1 TO
 19 STEP 3 :: DISPLAY AT(I+2,J):A$ :: J=
J+C :: NEXT I
1870 RETURN
1880 ! ADDITION
1890 Z(1)=INT(RND*B)
1900 IF G<4 THEN Z(2)=INT(RND*(B-Z(1)))E
LSE I=INT(Z(1)/T)*T+T :: Z(2)=INT(RND*(I
-Z(1)))+1
1910 IF RZ=1 THEN 1950
1920 CALL REL(Z(1)):: CALL REL(Z(2))
1930 IF T=0 THEN 1950
1940 IF ABS(INT((Z(1)+Z(2))/T))<ABS(INT(
Z(1)/T))THEN Z(1)=-Z(1)!BEREICHEINHALTUN
G BEI REL.ZAHLEN
1950 V$="+" :: Z(3)=Z(1)+Z(2):: Z$(3)=ST
R$(Z(3))
1960 RETURN
1970 ! SUBTRAKTION
1980 Z(1)=INT(RND*B)+1
1990 IF G<4 THEN Z(2)=INT(RND*Z(1))ELSE
I=Z(1)-INT(Z(1)/T)*T :: Z(2)=INT(RND*I)
2000 IF RZ=1 THEN 2040
2010 CALL REL(Z(1)):: CALL REL(Z(2))
2020 IF T=0 THEN 2040
2030 IF ABS(INT((Z(1)-Z(2))/T))>ABS(INT(
Z(1)/T))THEN Z(1)=-Z(1)
2040 V$="-" :: Z(3)=Z(1)-Z(2):: Z$(3)=ST
R$(Z(3))
2050 RETURN
2060 ! MULTIPLIKATION
2070 ON G GOTO 2090,2170,2210,2250
2080 ! KLEINES 1*1
2090 Z(1)=INT(RND*10)+1
2100 Z(2)=INT(RND*10)+1
2110 IF RZ=1 THEN 2130
2120 CALL REL(Z(1)):: CALL REL(Z(2))
2130 IF G2=1 OR B<150 THEN 2150
2140 IF Z(1)<10 OR Z(2)<10 THEN 2070 !GR
OSSE ZAHLEN BEI SCHRFTL RECHNEN
2150 V$="*" :: Z(3)=Z(1)*Z(2):: Z$(3)=ST
R$(Z(3)):: RETURN
2160 ! GROSSES 1*1
2170 Z(1)=INT(RND*20)+1 :: J=INT(Z(1)/10
)+1
2180 Z(2)=INT(RND*20/J)+1
2190 GOTO 2110
2200 !UNBESCHRAENKT MULTIPL
2210 Z(1)=INT(RND*B/3)+1
2220 Z(2)=INT(RND*B/Z(1))+1
2230 GOTO 2110
2240 !BESTIMMTES 1*1
2250 I=INT(RND*2)+1 :: Z(I)=C
2260 Z(3-I)=INT(RND*10)+1 :: GOTO 2110
2270 ! DIVISION
2280 R=0 :: ON G GOTO 2300,2380,2420,246
0
2290 !KLEINES 1*1
2300 Z(2)=INT(RND*10)+1
2310 Z(3)=INT(RND*10)+1
2320 IF RZ=1 THEN 2340
2330 CALL REL(Z(2)):: CALL REL(Z(3))
2340 Z(1)=Z(2)*Z(3):: Z$(3)=STR$(Z(3))
2350 IF G1=2 THEN R=INT(RND*ABS(Z(2)))::
 Z(1)=Z(1)+R*SGN(Z(1)):: Z$(3)=Z$(3)&"+R
EST "&STR$(R)
2360 V$=":" :: RETURN
2370 !GROSSES 1*1
```

Scan: S. Höltgen


```
2380 Z(2)=INT(RND*20)+1 :: J=INT(Z(2)/10
)+1
2390 Z(3)=INT(RND*20/J)+1
2400 GOTO 2320
2410 !UNBESCHRAENKT TEILEN
2420 Z(2)=INT(RND*B/3)+1
2430 Z(3)=INT(RND*B/Z(1))+1
2440 GOTO 2320
2450 !BESTIMMTES 1*1
2460 I=INT(RND*2)+2 :: Z(I)=C
2470 Z(5-I)=INT(RND*10)+1 :: GOTO 2320
2480 SUB REL(Z)
2490 I=INT(RND*10)+1
2500 IF I<6 THEN Z=-Z
2510 SUBEND
2520 SUB KLAMMER(Z,Z$)
2530 IF Z<0 THEN Z$="("&STR$(Z)&")"
2540 SUBEND
2550 SUB EDIT(A,P,Z$)
2560 CALL KEY(0,K,S)
2570 CALL GCHAR(10+A,P,J)
2580 CALL HCHAR(10+A,P,30)
2590 CALL HCHAR(10+A,P,J)
2600 IF S<1 THEN 2560
2610 IF K=8 OR K=83 THEN 2660
2620 IF K=9 OR K=68 THEN P=MIN(30,P+1)::
 GOTO 2560
2630 IF K=32 OR K=45 OR K=43 OR K=40 OR
K=41 THEN 2650
2640 IF K=13 THEN 2680
2650 IF K<>30 THEN CALL HCHAR(10+A,P,K)E
LSE CALL HCHAR(10+A,P,32)
2660 P=MAX(25,P-1)
2670 GOTO 2560
2680 Z$=""
2690 FOR S=1 TO 7 :: CALL GCHAR(10+A,23+
S,J)
2700 IF J<>32 AND J<>61 THEN Z$=Z$&CHR$(
J)
2710 NEXT S
2720 SUBEND
2730 SUB CHECK(Z$,Z1$,F,G2,D,L1,A$,P)
2740 IF Z$=Z1$ THEN F=0 :: GOTO 2840
2750 H$="("&Z$&")" :: IF Z1$=H$ THEN DIS
PLAY AT(19,11):Z$;" EINKLAMMERN!" :: DIS
PLAY AT(20,16):"SONST RICHTIG" :: GOTO 2
770
2760 CALL SOUND(100,138,4):: CALL SOUND(
200,110,5):: DISPLAY AT(21,11):"NICHT RI
CHTIG"
2770 F=F+1 :: IF F<3 THEN DISPLAY AT(22,
11):"BITTE NOCH EINMAL!" :: CALL WART ::
 GOTO 2840
2780 DISPLAY AT(13+D,11)SIZE(8):"LOESUNG
:"
2790 IF G2=1 THEN D=D+2
2800 DISPLAY AT(13+D,29-LEN(A$)):A$
2810 FOR J=1 TO 3 :: DISPLAY AT(13+D,P)S
IZE(L1)BEEP:Z1$ :: FOR I=1 TO 50 :: NEXT
 I
2820 CALL HCHAR(13+D,P+2,32,L1):: FOR I=
1 TO 50 :: NEXT I :: NEXT J
2830 DISPLAY AT(13+D,P)SIZE(L1)BEEP:Z1$
:: CALL HCHAR(21,11,32,21):: CALL WART
2840 SUBEND
2850 SUB WART
2860 DISPLAY AT(24,22):">ENTER<"
2870 CALL KEY(0,K,S):: IF S=0 THEN 2870
2880 SUBEND
```

Macro-Assembler/ Editor (Listing)

Nachdem in der letzten Ausgabe von HC eine umfangreiche Beschreibung dieses professionellen Werkzeuges zum Erstellen von Maschinenprogrammen für den C 64 war, folgt jetzt das komplette Listing mit Tips zur korrekten Eingabe

Um die 9,5 KByte des Assemblers in den Speicher zu bringen, wurde das Programm in DATAs gepackt, die von einem kleinen BASIC-Programm gelesen und in den Speicher gepoket werden. Leider passen die DATAs für 9,5 KByte Maschinen-Code nicht auf einmal in den Arbeitsspeicher. Das Ladeprogramm wurde deshalb in zwei Teile aufgespalten, die nacheinander abgearbeitet werden müssen.

Es ist sicher nicht jedem gegeben, 9728 Zahlen auf Anhieb richtig einzugeben. Das Ladeprogramm prüft deshalb die einzelnen Zeilen auf die korrekte Anzahl der Statements, berechnet aus den Werten eine Prüfsumme und ein Parity-Byte. Eingegebene Fehler sollten damit halbwegs sicher erkannt werden. Geben Sie bitte die DATAs wie abgedruckt ein. Spätestens bei der fast unvermeidlichen Fehlersuche werden Sie um eine saubere Eingabe froh sein. Noch etwas: Speichern Sie Ihre DATAs unbedingt vor jedem Probelauf ab!

Um die Suche nach der Ursache des Prüfsummenfehlers, der mit einiger Sicherheit auftritt, nicht ganz aussichtslos erscheinen zu lassen, enthält das Ladeprogramm eine Prüfsummenroutine, mit der Tippfehler aufgespürt werden können. Es sind dazu die Prüfsummen/Parity-Werte kleinerer Blöcke abgedruckt. Findet man einen fehlerhaften Block, so prüft man anschließend die darin enthaltenen nächstkleineren Blöcke. Man kann sich so bis auf den Bereich von vier Zeilen an den Fehler herantasten. Oft ist man nur beim Abtippen in die falsche Zeile geraten oder hat die Ziffern 8, 6, 3 oder 9 verwechselt.

In Teil 1 wird der vom Assembler belegte Bereich geschützt und die „untere" Hälfte des Programms in den Speicher geschrieben. Wenn im zweiten Programmblock kein Fehler auftrat, wird der Assembler, der im Speicherbereich \$7A00 bis \$A000 liegt, als Maschinenprogramm abgespeichert. Der so abgespeicherte Assembler kann dann von dem noch einzugebenden Lader aufgerufen werden. Wenn der Assembler bereits geladen wurde, so kann er von BASIC aus mit "SYS31232" „kalt" gestartet werden, ein im Arbeitsspeicher befindlicher Quelltext wird dadurch gelöscht (kann mit UNNEW zurückgeholt werden). Mit "SYS31235" kann man den Assembler ohne Textverlust starten.

Wer das Monitor-Programm aus CHIP 1/84 besitzt, kann den Assembler mit:
.T 7A00,9FFF,0A00–
.ER 0A00,2FFF,7A00,9FFF,0A00–
an den Speicherbereich von \$0A00–\$3000 anpassen; der dabei gemeldete Fehler ist ohne Bedeutung.

Mit den folgenden Pokes lassen sich die Standardeinstellungen für Druckerkanal

Scan: S. Höltgen

Stichwort: BASIC (11)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

17. Häufige Fehler

BASIC Fehler-meldung	Beispiel	Erklärung
	D. Weitere Syntaxfehler	
SYNTAX ERROR	10 LET A> = 10	Nicht existierender Befehl. (A> = 10 ist eine mathematische Aussage, kein Befehl.)
BAD SUB-SCRIPT ERROR	20 X = ARC TG(15)	Der Computer interpretiert ARCTG als Namen eines Feldes, das jedoch nicht dimensioniert wurde. Gemeint war der Wert der Funktion ATN(15).
UNDEF'D FUNC-TION ERROR	30 C = FN A(4) 40 DEF FN A(X) = X ↑ 3-X	Falsche Reihenfolge: Die Definitionszeile muß vor der ersten Benutzung einer selbstdefinierten Funktion auftreten.
SYNTAX ERROR	50 LET MON = 1	Es wurde in einem Variablennamen ein reserviertes Wort (ON) benutzt.
	E. Eingabefehler	
REDO FROM START BZW. REENTER	10 INPUT A EINGABE: B	Eine Zahl wurde angefordert, ein Textzeichen eingegeben. Der Computer fordert erneut eine Zahl an.
EXTRA IGNORED	10 INPUT A, B, C EINGABE: 10, 20, 30, 40	Es wurden mehr Werte eingegeben als angefordert. Der Computer ignoriert überzählige Daten.
??	10 INPUT A, B, C EINGABE: 10, 11	Es wurden weniger Daten als erwartet eingegeben. Der Computer fordert die restlichen an.
	F. Einige typische Programmierfehler	
	100 DEF FN F(X)=X ↑ .205 200 Y = F(3)	Nicht die Funktion FN F wird aufgerufen, sondern das Feld F. Da manche Computer es zulassen, auch mit nicht vorher dimensionierten Feldern zu arbeiten, wird in Zeile 200 der Wert des Feldes auf den Speicherplatz Y geschrieben. (Z. B. APPLE II: Felder mit maximalem Index 10 benötigen keine vorhergehende DIM-Anweisung.) Eine Fehlermeldung kommt durch Zeilen 100 und 200 nicht zustande.
	2O A = 4*B	Anstelle der Ziffer 0 wurde bei der Zeilennummer der Buchstabe O benutzt. Es entsteht die Programmzeile 2, in der der Variablen OA der Wert 4*B zugewiesen wird. Die eventuell

Scan: S. Höltgen

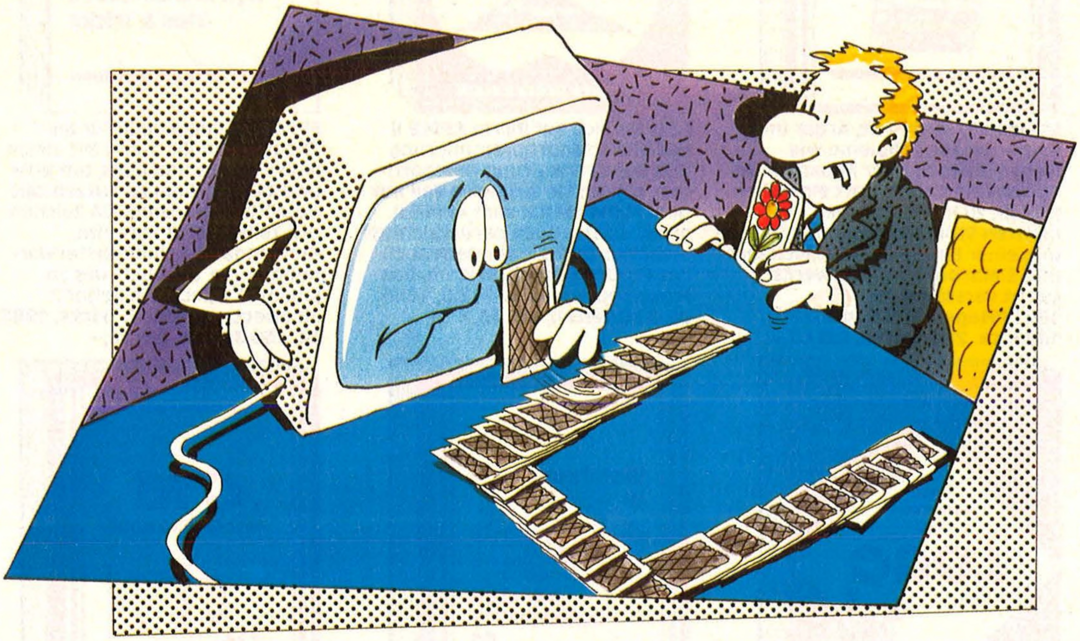
Remember

Lehrreich für jung und alt: zum Erfassen und Erkennen von Formen und zum Training des Gedächtnisses (TI-99/4A in der Grundversion)

Das Programm bietet die Möglichkeit, gegen den Computer oder einen menschlichen Gegner zu spielen. Bei der Wahl eines menschlichen Gegners kann man zwischen zwei Spielstufen wählen; entscheidet man sich für den Computer, stehen sechs Spiele mit vier Schwierigkeitsgraden zur Wahl.

Menschlicher Gegner:
Spiel 1 mit 20 Karten
Spiel 2 mit 42 Karten
Spieler 1 mit Joystick 1; Spieler 2 mit Joystick 2

Computer als Gegner:
Spiel 1–3 mit 20 Karten
Spiel 4–6 mit 42 Karten
Spieler mit Joystick 1; wobei der Computer bei den Spielen
1 und 4 nur zufällig zieht,
2 und 5 sich seine eigenen aufgedeckten Karten merkt,



Zeilen	Kommentar
Hauptprogramm	
30–270	Titelbild
280–600	Bild für die Auswahl der Spielvariation
610–960	Auswahl der Spielvariation
970–1310	Änderung des Zeichensatzes, Spielfeldzeichnung sowie allgemeine Vorbereitungen (Variablenzuordnungen und Farbänderungen)
1320–1550	Mischen und Austeilen der Karten
1560–2360	Zug der (des) Spieler(s)
2370–3240	Zug des Computers
3250–3530	Ende des Spiels mit Ermittlung des Gewinners
Unterprogramme	
3540–3780	Speicherung der aufgedeckten Karte in COM1$; der Position der Karte in COM2$
3790–3850	Löschen des abgespeicherten Kartenpaares (falls im Speicher vorhanden) bei Aufdecken zwei identischer Karten
3860–3930	Systematische Suche der zweiten aufzudeckenden Karte des Computers (in dieses Unterprogramm springt der Computer nur, wenn er fünfmal hintereinander eine zufällige Position auf dem Spielfeld wählt, auf der sich keine Karte mehr befindet)
3940–4000	Wie in den Zeilen 3860–3930, jedoch für die erste aufzudeckende

Zeilen	Kommentar
	Karte des Computers
4010–4050	Sichtbarmachen eines quadratischen Feldes, welches aus vier Zeichen besteht (u. a. Aufdecken einer Karte)
4060–4080	Zudecken einer Karte
4090–4230	Punkteauswertung (entweder vom Computer oder vom zweiten Spieler)
4240–4380	Punkteauswertung (vom ersten Spieler)
4390–4450	Zurücksetzen der Positionsvariablen
4460–4520	Sound-Unterprogramm
Daten	
4530–4550	Daten für Sonderzeichen im Titelbild
4560–4590	Daten für Sonderzeichen bei der Spielauswahl
4600–4610	Daten für Positionierung der Sonderzeichen bei der Spielauswahl
4620–4750	Daten für Sonderzeichen der Spielkarten
4760–4770	Daten für Breite und Höhe des Spielfeldes sowie Anzahl der Karten
4780–4790	Daten für Sonderzeichen („Digitalzahlen“)
4800–4810	Daten für Sound-Unterprogramm
4820	Daten für Sonderzeichen (Joystick)
4830	Anweisung, auf Kleinschrift umzuschalten, bevor man spielt

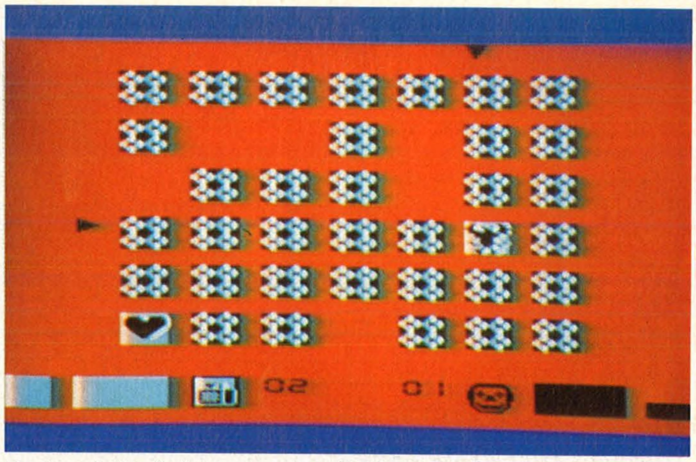
3 und 6 sich alle aufgedeckten Karten merkt.

Die Abfrage der gewünschten Spielvariation erfolgt jeweils am Anfang eines Spiels und wird mit Hilfe des Joysticks beantwortet, wobei jegliche Eingabe mit einem Druck auf die Feuertaste beendet werden muß.

Die Auswahl einer Karte, die aufgedeckt werden soll, erfolgt durch Einstellen der beiden Pfeile auf die gewünschte Position und anschließendes Drücken auf die Feuertaste. Bei der Programmierung wurde weitgehend auf Dimensionierungen verzichtet (die Speicherung von 42 Karten hätte allein schon mehr als ein Drittel des Speicherplatzes belegt) und sehr viel mit Strings gearbeitet. Das ist zwar weniger elegant und schwerer zu verstehen, es geschah jedoch aufgrund von Speicherplatzproblemen. Der Computer speichert übrigens die aufgedeckten Karten in den beiden Strings COM1$ und COM2$.

Nun viel Spaß und Unterhaltung mit „Remember"!

Jörg Schuntermann



Pech gehabt: kein passendes Paar

```
30 CALL CLEAR
40 CALL SCREEN(2)
50 RESTORE 4530
60 FOR A=48 TO 62
70 READ A$
80 CALL CHAR(A,A$)
90 NEXT A
100 PRINT "   J.S.- S O F T W A R E
                                PRES
ENTS":::
110 PRINT "             **************
        *M E M O R Y*               ****
*********":::
120 PRINT "       410 COPYRIGHT BY
  41412                     4141453 J.SC
HUNTERMANN"
130 PRINT " <:9; 63
= 9:963                        9: 973 PRES
S FIREBUTTON"
140 PRINT "     9: 3 FOR BEGINNING
    98":::
150 CALL HCHAR(1,1,62,64)
160 CALL VCHAR(1,1,62,48)
170 CALL VCHAR(1,31,62,48)
180 CALL HCHAR(23,1,62,64)
190 FOR A=2 TO 8
200 CALL COLOR(A,16,2)
210 NEXT A
220 RANDOMIZE
230 DEF SP=INT(CO/10)+1
240 CALL KEY(1,K,S)
250 IF S<>0 THEN 280
260 CALL KEY(2,K,S)
270 IF S=0 THEN 240
280 CALL HCHAR(1,1,152,768)
290 RESTORE 4560
300 FOR A=137 TO 142
310 READ A$
320 CALL CHAR(A,A$)
330 NEXT A
340 FOR A=144 TO 158
350 READ A$
360 CALL CHAR(A,A$)
370 NEXT A
380 RESTORE 4780
390 FOR A=48 TO 54
400 READ A$
410 CALL CHAR(A,A$)
420 NEXT A
430 CALL SCREEN(5)
440 CALL COLOR(3,9,9)
450 CALL COLOR(14,9,9)
460 CALL COLOR(15,9,9)
470 CALL COLOR(16,9,9)
480 RESTORE 4600
490 FOR A=1 TO 6
500 READ B
510 CALL HCHAR(3,2*A+9,B)
520 NEXT A
530 FOR A=1 TO 8
540 READ B,C,D
550 CALL HCHAR(B,C,D)
560 NEXT A
570 CALL COLOR(3,8,9)
580 CALL COLOR(14,2,16)
590 CALL COLOR(15,2,9)
600 CALL COLOR(16,2,9)
610 C=0
620 V=6
630 IF C<>0 THEN 680
640 H=6
650 D=9
660 E=3
670 GOTO 710
680 H=21
690 D=16-GEGNER*8
700 E=2
710 CALL HCHAR(V,H,158)
720 CALL KEY(1,K,S)
730 IF S<>0 THEN 850
740 CALL KEY(2,K,S)
750 IF S<>0 THEN 850
760 CALL JOYST(1,K,S)
770 IF S<>0 THEN 800
780 CALL JOYST(2,K,S)
790 IF S=0 THEN 720
800 IF (V-S/4*E<6)+(V-S/4*E>D)<0 THEN 72
0
810 CALL HCHAR(V,H,152)
820 V=V-S/4*E
830 CALL HCHAR(V,H,158)
840 GOTO 720
850 CALL SOUND(200,990,5,989,5,988,5)
860 IF C<>0 THEN 950
870 GEGNER=ABS(V=9)
880 C=1
890 FOR A=1 TO 6-GEGNER*4
900 READ B
910 CALL HCHAR(4+A*2,23,A+48)
920 CALL HCHAR(4+A*2,25,B)
930 NEXT A
940 GOTO 620
950 LEV=V/2-2+GEGNER*2
960 CALL HCHAR(1,1,152,768)
970 FOR A=1 TO 13
980 CALL COLOR(A,2,16)
990 NEXT A
1000 FOR A=32 TO 132 STEP 5
1010 CALL CHAR(A,"E7C3A51818A5C3E7")
1020 NEXT A
1030 CALL HCHAR(22,5,141,4)
1040 CALL HCHAR(23,5,141,4)
1050 VE=22
1060 HO=10
1070 KAR=GEGNER*7+136
1080 GOSUB 4010
1090 HO=22
1100 KAR=143
1110 GOSUB 4010
1120 CALL HCHAR(22,25,142,4)
1130 CALL HCHAR(23,25,142,4)
1140 PUNKTE(2)=0
1150 GOSUB 4090
1160 PUNKTE(1)=0
1170 GOSUB 4240
1180 RESTORE 4620
1190 FOR A=33 TO 133 STEP 5
1200 FOR B=A TO A+3
```


Scan: S. Höltgen

Scan: S. Höltgen


```
1210 READ A$
1220 CALL CHAR(B,A$)
1230 NEXT B
1240 NEXT A
1250 FOR A=156 TO 157
1260 READ A$
1270 CALL CHAR(A,A$)
1280 NEXT A
1290 CO=2
1300 COM1$=""
1310 COM2$=""
1320 K$="    %%**//4499>>CCHHMMRRWW\\aaff
kkppuuzz||}}"
1330 ON LEV GOTO 1340,1340,1340,1360,136
0,1360
1340 RESTORE 4760
1350 GOTO 1370
1360 RESTORE 4770
1370 READ HO1,HO2,VE1,VE2,ANZ
1380 KARTENZAHL=ANZ
1390 FOR HOR=HO1 TO HO2 STEP 3
1400 FOR VER=VE1 TO VE2 STEP 3
1410 A=INT(RND*ANZ)+1
1420 KAR$=SEG$(K$,A+1,1)
1430 K$=SEG$(K$,1,A)&SEG$(K$,A+2,ANZ-A)
1440 ANZ=ANZ-1
1450 IF KAR$<>"|" THEN 1480
1460 B=127
1470 GOTO 1520
1480 IF KAR$<>"}" THEN 1510
1490 B=132
1500 GOTO 1520
1510 B=ASC(KAR$)
1520 CALL VCHAR(VER,HOR,B,2)
1530 CALL VCHAR(VER,HOR+1,B,2)
1540 NEXT VER
1550 NEXT HOR
1560 CALL COLOR(16,CO,9)
1570 CALL HCHAR(VE1,HO1-2,156)
1580 CALL HCHAR(VE1-2,HO1,157)
1590 V1=VE1
1600 H1=HO1
1610 CALL JOYST(SP,H,V)
1620 IF H<>0 THEN 1750
1630 IF V<>0 THEN 1660
1640 CALL KEY(SP,K,S)
1650 IF S=0 THEN 1610 ELSE 1840
1660 IF V<0 THEN 1690
1670 V=V-1
1680 GOTO 1700
1690 V=V+1
1700 IF (V1-V<VE1)+(V1-V>VE2)<0 THEN 161
0
1710 CALL HCHAR(V1,HO1-2,152)
1720 V1=V1-V
1730 CALL HCHAR(V1,HO1-2,156)
1740 GOTO 1610
1750 IF H<0 THEN 1780
1760 H=H-1
1770 GOTO 1790
1780 H=H+1
1790 IF (H1+H<HO1)+(H1+H>HO2)<0 THEN 161
0
1800 CALL HCHAR(VE1-2,H1,152)
1810 H1=H1+H
1820 CALL HCHAR(VE1-2,H1,157)
1830 GOTO 1610
1840 CALL SOUND(-200,1387,5,1397,10,1407
,15)
1850 VE=V1
1860 HO=H1
1870 CALL GCHAR(VE,HO,KAR)
1880 IF KAR=152 THEN 1610
1890 IF KARTE>0 THEN 1950
1900 KARTE1=KAR
1910 VESPEICHER=VE
1920 HOSPEICHER=HO
1930 KARTE=KARTE+1
1940 GOTO 1980
1950 IF (VE=VESPEICHER)+(HO=HOSPEICHER)=
-2 THEN 1610
1960 KARTE2=KAR
1970 KARTE=KARTE+1
1980 GOSUB 4010
1990 IF (LEV=3)+(LEV=6)>=0 THEN 2010
2000 GOSUB 3540
2010 IF KARTE<2 THEN 1610
2020 KARTE=0
2030 IF KARTE1<>KARTE2 THEN 2110
2040 ON SP GOTO 2050,2070
2050 RESTORE 4810
2060 GOTO 2080
2070 RESTORE 4800
2080 GOSUB 4460
2090 ZU=152
2100 GOTO 2140
2110 FOR A=1 TO 400+((INT(LEV/3)=LEV/3)*
200)
2120 NEXT A
2130 ZU=KARTE2
2140 GOSUB 4060
2150 IF KARTE1=KARTE2 THEN 2170
2160 ZU=KARTE1
2170 VE=VESPEICHER
2180 HO=HOSPEICHER
2190 GOSUB 4060
2200 IF KARTE1<>KARTE2 THEN 2310
2210 KARTENZAHL=KARTENZAHL-2
2220 PUNKTE(SP)=PUNKTE(SP)+1
2230 IF SP=1 THEN 2260
2240 GOSUB 4090
2250 GOTO 2270
2260 GOSUB 4240
2270 IF KARTENZAHL=0 THEN 3250
2280 GOSUB 3790
2290 GOSUB 4390
2300 GOTO 1610
2310 CALL VCHAR(VE1,HO1-2,152,16)
2320 CALL HCHAR(VE1-2,HO1,152,19)
2330 IF GEGNER=0 THEN 2360
2340 CO=ABS((CO=2))*14+2
2350 GOTO 1560
2360 IF (LEV=1)+(LEV=4)<0 THEN 2430
2370 FOR B=0 TO LEN(COM1$)-3 STEP 3
2380 SPEICHER$=SEG$(COM1$,B+1,3)
2390 FOR C=3+B TO LEN(COM1$)-3 STEP 3
2400 IF SPEICHER$=SEG$(COM1$,C+1,3)THEN
2770
2410 NEXT C
2420 NEXT B
2430 Z=0
2440 VS=VE1+3*INT(RND*(VE2+3-VE1)/3)
2450 HS=HO1+3*INT(RND*(HO2+3-HO1)/3)
2460 CALL GCHAR(VS,HS,COMKARTE1)
2470 IF COMKARTE1<>152 THEN 2510
2480 Z=Z+1
2490 IF Z<5 THEN 2440
2500 GOSUB 3940
2510 VE=VS
2520 HO=HS
2530 KAR=COMKARTE1
2540 GOSUB 4010
2550 VARIA=1
2560 IF (LEV=1)+(LEV=4)<0 THEN 2610
2570 FOR B=0 TO LEN(COM1$)-3 STEP 3
2580 IF VAL(SEG$(COM1$,B+1,3))<>COMKARTE
1 THEN 2600
2590 GOTO 2770
2600 NEXT B
2610 IF (LEV=1)+(LEV=4)<0 THEN 2630
2620 GOSUB 3540
2630 Z=0
2640 VE=VE1+3*INT(RND*(VE2+3-VE1)/3)
2650 HO=HO1+3*INT(RND*(HO2+3-HO1)/3)
2660 IF (VE=VS)+(HO=HS)=-2 THEN 2630
2670 CALL GCHAR(VE,HO,COMKARTE2)
2680 IF COMKARTE2<>152 THEN 2720
2690 Z=Z+1
2700 IF Z<5 THEN 2640
2710 GOSUB 3860
2720 KAR=COMKARTE2
2730 GOSUB 4010
2740 IF (LEV=1)+(LEV=4)<0 THEN 2760
2750 GOSUB 3540
2760 GOTO 2990
2770 VC=VAL(SEG$(COM2$,B*4/3+1,2))
2780 HC=VAL(SEG$(COM2$,B*4/3+3,2))
2790 IF (VS=VC)+(HS=HC)=-2 THEN 2630
2800 IF VARIA<>1 THEN 2840
2810 CALL GCHAR(VC,HC,COMKARTE2)
2820 KAR=COMKARTE2
2830 GOTO 2860
2840 CALL GCHAR(VC,HC,COMKARTE1)
2850 KAR=COMKARTE1
2860 VE=VC
2870 HO=HC
```

Scan: S. Höltgen

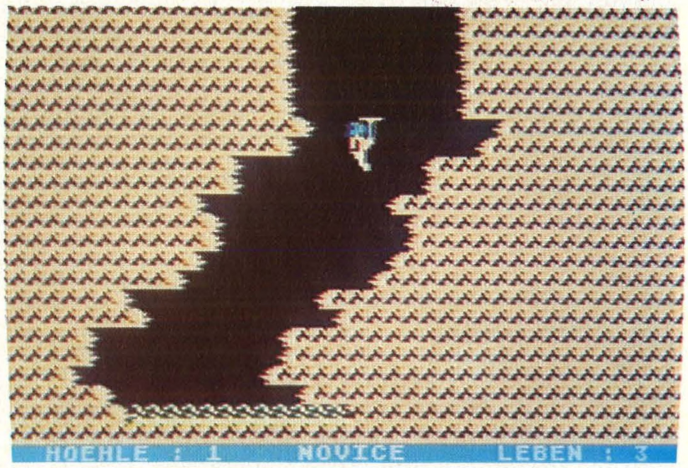

```
2880 GOSUB 4010
2890 IF (LEV=1)+(LEV=4)<0 THEN 2910
2900 GOSUB 3540
2910 IF VARIA=1 THEN 2990
2920 VE=VAL(SEG$(COM2$,C*4/3+1,2))
2930 HO=VAL(SEG$(COM2$,C*4/3+3,2))
2940 CALL GCHAR(VE,HO,COMKARTE2)
2950 KAR=COMKARTE2
2960 GOSUB 4010
2970 IF (LEV=1)+(LEV=4)<0 THEN 2990
2980 GOSUB 3540
2990 IF COMKARTE1<>COMKARTE2 THEN 3040
3000 RESTORE 4800
3010 GOSUB 4460
3020 ZU=152
3030 GOTO 3070
3040 FOR A=1 TO 400+((INT(LEV/3)=LEV/3)*
200)
3050 NEXT A
3060 ZU=COMKARTE2
3070 GOSUB 4060
3080 IF COMKARTE1=COMKARTE2 THEN 3100
3090 ZU=COMKARTE1
3100 IF VARIA=1 THEN 3140
3110 VE=VC
3120 HO=HC
3130 GOTO 3170
3140 VE=VS
3150 HO=HS
3160 VARIA=0
3170 GOSUB 4060
3180 GOSUB 4390
3190 IF COMKARTE1<>COMKARTE2 THEN 1570
3200 GOSUB 3790
3210 KARTENZAHL=KARTENZAHL-2
3220 PUNKTE(2)=PUNKTE(2)+1
3230 GOSUB 4090
3240 IF KARTENZAHL>0 THEN 2360
3250 CALL HCHAR(VE1-2,HO1,152,19)
3260 CALL VCHAR(VE1,HO1-2,152,16)
3270 FOR A=1 TO 7
3280 IF PUNKTE(1)=PUNKTE(2)THEN 3320
3290 CALL CHAR(141+ABS(PUNKTE(1)>PUNKTE(
2)),"")
3300 CALL CHAR(141+ABS(PUNKTE(1)>PUNKTE(
2)),"FFFFFFFFFFFFFFFF")
3310 GOTO 3360
3320 CALL CHAR(141,"")
3330 CALL CHAR(142,"")
3340 CALL CHAR(141,"FFFFFFFFFFFFFFFF")
3350 CALL CHAR(142,"FFFFFFFFFFFFFFFF")
3360 NEXT A
3370 IF PUNKTE(1)>PUNKTE(2)THEN 3390
3380 CALL CHAR(141,"")
3390 RESTORE 4820
3400 FOR A=153 TO 156
3410 READ A$
3420 CALL CHAR(A,A$)
3430 NEXT A
3440 CALL COLOR(16,2,9)
3450 VE=11
3460 HO=16
3470 KAR=152
3480 GOSUB 4010
3490 CALL SOUND(100,1397,0)
3500 CALL KEY(1,K,S)
3510 IF S<>0 THEN 280
3520 CALL KEY(2,K,S)
3530 IF S=0 THEN 3500 ELSE 280
3540 GOTO 3700
3550 IF KAR>99 THEN 3580
3560 KAR$="0"&STR$(KAR)
3570 GOTO 3590
3580 KAR$=STR$(KAR)
3590 COM1$=COM1$&KAR$
3600 IF VE>9 THEN 3630
3610 VE$="0"&STR$(VE)
3620 GOTO 3640
3630 VE$=STR$(VE)
3640 IF HO>9 THEN 3670
3650 HO$="0"&STR$(HO)
3660 GOTO 3680
3670 HO$=STR$(HO)
3680 COM2$=COM2$&VE$&HO$
3690 RETURN
3700 IF LEN(COM1$)=0 THEN 3550
3710 KAR$=STR$(KAR)
3720 FOR Z=0 TO LEN(COM1$)-3 STEP 3
3730 IF VAL(KAR$)=VAL(SEG$(COM1$,Z+1,3))
THEN 3760
3740 NEXT Z
3750 GOTO 3550
3760 IF (VE=VAL(SEG$(COM2$,Z*4/3+1,2)))+
(HO=VAL(SEG$(COM2$,Z*4/3+3,2)))=-2 THEN
3690
3770 IF Z<LEN(COM1$)-3 THEN 3740 ELSE 35
50
3780 GOSUB 4460
3790 FOR A=0 TO LEN(COM1$)-3 STEP 3
3800 IF KAR<>VAL(SEG$(COM1$,A+1,3))THEN
3840
3810 COM1$=SEG$(COM1$,1,A)&SEG$(COM1$,A+
4,LEN(COM1$)-A)
3820 COM2$=SEG$(COM2$,1,A*4/3)&SEG$(COM2
$,A*4/3+5,LEN(COM2$)-A*4/3)
3830 GOTO 3790
3840 NEXT A
3850 RETURN
3860 FOR HO=HO1 TO HO2 STEP 3
3870 FOR VE=VE1 TO VE2 STEP 3
3880 CALL GCHAR(VE,HO,COMKARTE2)
3890 IF COMKARTE2=152 THEN 3910
3900 IF (VE=VS)+(HO=HS)<>-2 THEN 3930
3910 NEXT VE
3920 NEXT HO
3930 RETURN
3940 FOR HS=HO1 TO HO2 STEP 3
3950 FOR VS=VE1 TO VE2 STEP 3
3960 CALL GCHAR(VS,HS,COMKARTE1)
3970 IF COMKARTE1<>152 THEN 4000
3980 NEXT VS
3990 NEXT HS
4000 RETURN
4010 CALL HCHAR(VE,HO,KAR+1)
4020 CALL HCHAR(VE+1,HO,KAR+2)
4030 CALL HCHAR(VE,HO+1,KAR+3)
4040 CALL HCHAR(VE+1,HO+1,KAR+4)
4050 RETURN
4060 CALL VCHAR(VE,HO,ZU,2)
4070 CALL VCHAR(VE,HO+1,ZU,2)
4080 RETURN
4090 TEILER=10
4100 CALL HCHAR(22,13,152,2)
4110 RESTORE 4780
4120 FOR A=0 TO INT(PUNKTE(2)/TEILER)-IN
T(PUNKTE(2)/10)*(TEILER-10)/(-9)*10
4130 READ A$
4140 NEXT A
4150 CALL CHAR(148+INT(TEILER/10),A$)
4160 IF TEILER=1 THEN 4190
4170 TEILER=1
4180 GOTO 4110
4190 IF PUNKTE(2)=0 THEN 4210
4200 CALL HCHAR(24-PUNKTE(2),2,141,2)
4210 CALL HCHAR(22,13,149)
4220 CALL HCHAR(22,14,148)
4230 RETURN
4240 TEILER=10
4250 CALL HCHAR(22,19,152,2)
4260 RESTORE 4780
4270 FOR A=0 TO INT(PUNKTE(1)/TEILER)-IN
T(PUNKTE(1)/10)*(TEILER-10)/(-9)*10
4280 READ A$
4290 NEXT A
4300 CALL CHAR(150+INT(TEILER/10),A$)
4310 IF TEILER=1 THEN 4340
4320 TEILER=1
4330 GOTO 4260
4340 IF PUNKTE(1)=0 THEN 4360
4350 CALL HCHAR(24-PUNKTE(1),30,142,2)
4360 CALL HCHAR(22,19,151)
4370 CALL HCHAR(22,20,150)
4380 RETURN
4390 VS=0
4400 HS=0
4410 VC=0
4420 HC=0
4430 VE=0
4440 HO=0
4450 RETURN
4460 FOR TON=1 TO 4
4470 READ FREQUENZ,DAUER
4480 FOR NOISE=0 TO 10 STEP 2
4490 CALL SOUND(-DAUER*15,FREQUENZ,NOISE
)
4500 NEXT NOISE
```

Scan: S. Höltgen


```
4510 NEXT TON
4520 RETURN
4530 DATA C0300C03030D31C1,030C30C,01010
10101010101,8181818181818181,00000000030
C30C,030C30C08080808
4540 DATA 8080808080808080,808080808,838C
B0C,C0300C03,00000000C0300C03,030C30C0C0
300C03,C0B08C838080808
4550 DATA 80808080C0300C03,FFFFFFFFFFFFF
FFF
4560 DATA FF808080878280BF,AAB5AAB5AABF8
0FF,FF01010141410DD,DD5DDD5DDDDD01FF,,F
FFFFFFFFFFFFFFF
4570 DATA 001F3F6040C8D4C9,C1C340504F271
00F,00F8FC0602132B93,83C3020AF2E408F,007
F4941416161,007F407840607F
4580 DATA 007F414141617F,007F417F426161,
,0041417F080C0C,0000000007020707,000E0E0
41F1F1F1F,1E1E0C3F3F3F3F
4590 DATA 7E7E18FFFFFFFFFF,0040707C7F7C7
04
4600 DATA 148,149,148,150,151,153,6,8,13
7,7,8,138,6,9,139,7,9,140,9,8,144,10,8,1
45,9,9,146
4610 DATA 10,9,147,154,155,156,155,156,1
57
4620 DATA FFC0C0C7FEFDFDF3,F3F3CFCF3F00F
FFC,FF0303E37FBFBFCF,CFCFF3F3FC00FF3F,00
3F000F10100F,000F1010100807
4630 DATA 00C040FC4040F844,44F84040408,0
01806317DEF7F07,1F7DFBF7EF7E1C,0018608CB
EF7FEE,F8BEDFEFF77E38
4640 DATA 000C121101433F1D,1F1F3D1F1F6F8
301,0030488880C2FCB8,F8F8BCF8F8F6C18,000
03C7E7F7F7F3F,1F0F070301
4650 DATA 00003C7EFEFEFEFC,F8F0E0C08,F0F
0F0F00F0F0F0F,F0F0F0F00F0F0F0F,F0F0F0F00
F0F0F0F,F0F0F0F00F0F0F0F
4660 DATA 404040404047F,41412222141408,
7C4444444444FC,FC40407C40407C,0040301C0E
070103,070D1F1F0F030101
4670 DATA 0002060E1C38B0F,F8FCFCFCFCF8F0
E,070A1F323F321F0A,0703010101010101,E050
F84CFC4CF85,E0C08080808080808
4680 DATA 0000FE1012101212,12121212,0000
EEAAEE222222,222222EE,00547C6C282A3F3F,3
F2E2C3D3D3D3D,002A3E361454FCFC
4690 DATA FC7434BCBCBCBC,000000000070F1D3
F,E93F0F03,00000000E0F0B8FC,97FCF0C,0005
236919398543,FF43853919692305
4700 DATA 0040882C30384284,FF844238302C8
84,001F101013392F38,7B607F067E333,00F808
08C89CF41C,DE06FE607ECC0C
4710 DATA 003062E1797F0F0F,0303671B01020
C04,000C4687CEFEF0F,C0C0E6D88040302,001F
202728292A2A,2A292827201F
4720 DATA 00F804E414945494,14E404F800FC,
0F10274893A4A9AA,AAAAA9A4934827,F008E412
C9259555,9525C912E408F
4730 DATA FEFEFAF2E2C28202,FFFF7F1F0F0E0
C18,BF9F8F878381808,FFFCF0C0,04083010780
E0602,37FDFF0707020301
4740 DATA 20100C081E706040,ECBFFFE0E040C
08,1C3E7F7F7F604F0507,0507020302030101,3
0787C7C7C0CE840C0,40C08080808
4750 DATA BBB99CCFE7F3F8FF,FFF8F3E7CF9CB
9BB,DD9D39F3E7CF1FFF,FF1FCFE7F3399DDD,00
40707C7F7C704,00FE7C7C3838101
4760 DATA 10,22,6,15,20
4770 DATA 7,25,3,18,42
4780 DATA 003C42420042423C,000202020002O
202,003C02023C40403C,003C02023C02023C,00
0042423C0202,003C40403C02023C
4790 DATA 003C40403C42423C,003C020200020
2,003C42423C42423C,003C42423C02023C
4800 DATA 195,5,155,5,174,5,116,20
4810 DATA 116,5,174,5,195,5,155,20
4820 DATA 0121212121A97121,0377FFFFFFFFF
F7F,8080808080808080,C0E0FFFFFFFFFFFE
4830 REM  AUF KLEINSCHRIFT              UM
SCHALTEN                    (alpha-lock-ta
ste              loesen)
```

Höhlenflieger

Angeschnallt an einen kleinen Hubschrauber versucht der Held dieses Spieles, unbeschadet bis zur letzten von bis zu 20 Höhlen zu fliegen (Atari 800XL)



Nach RUN dauert es eine Weile, bis das Titelbild erscheint. Wenn die Starttaste gedrückt wird, beginnt das Spiel. Ziel ist es, die Spielfigur so durch die Höhle zu steuern, daß keine Wand gerammt wird. Ausgerüstet ist sie lediglich mit einem kleinen Hubschrauber, der auf seinen Rücken geschnallt ist und acht Säurebomben pro Bildschirm, die bei Bedarf abgeworfen werden können (Joystick nach unten), um Gesteinsbrocken wegzuätzen, z.B. wenn der Durchgang durch Barrikaden versperrt wird.

Gesteuert wird die Figur mit dem Joystick links und rechts, mit dem Feuerknopf rauf und runter. Ist der Feuerknopf gedrückt, so werden die Rotoren des Hubschraubers beschleunigt, und die Figur steigt aufwärts. Bei nicht gedrücktem Feuerknopf sackt sie ab. Wie viele Höhlen man durchfliegen muß, hängt vom gewählten Schwierigkeitsgrad ab:
bei Novice 10 Höhlen
bei Skilled 15 Höhlen
bei Expert 20 Höhlen.

Hat man alle Höhlenabschnitte durchflogen, so gelangt man schließlich zum Höhlenende. Hier gilt es nur noch, auf dem unteren begrasten Höhlenboden zu landen. Ist diese letzte Hürde auch überwunden, so kann das Spiel mit der Starttaste wieder neu begonnen werden. *Oliver Cyranka*

Scan: S. Höltgen

```
5005 BORDER 1
5010 FOR i=6 TO 25: PRINT PAPER
4;AT 15,i;"C": NEXT i
5020 PRINT AT 8,14;"DE": PRINT A
T 14,20;"FG"
5030 PRINT INK 7; PAPER 1; FLASH
 1;AT 0,0;"      O P A   S I N C
L A I R     "
5040 PRINT AT 11,0; INVERSE 1;"v
on Stephan Ruhmschoettel (1984)"
5100 PRINT AT 21,6; PAPER 2; INK
 6; FLASH 1;"Bitte Taste druecke
n"
5120 PAUSE 0: PAPER 7: CLS : PRI
NT "           ANLEITUNG
               =========
      Opa Sinclair ist mal wieder
mit seinem Gelaendewagen unterwe
gs. Doch wie so oft hat er seine
      Brille verloren. Zu allem Ue
ber-fluss faehrt er auch noch au
f     einer Bruecke, die noch nich
t     fertiggebaut ist.
      Sie muessen nun mit dem Ret-
      tungshubschrauber durch abwe
rfenvon Brueckenteilen verhinder
n,    dass Opa Sinclair ins Wasser
      faellt.
      Sollte das doch mal vorkomme
n,sowird er vom Hubschrauber ger
et- tet.(Wie oft haengt vom Schw
ie- rigkeitsgrad ab, wie auch di
e Punkte fuer Brueckenteile und
die fertige Bruecke"
5130 PRINT AT 21,0;"Taste drueck
en !": PAUSE 0: CLS
5140 PRINT "Das Abwerfen des Bru
eckenteils   wird mit der SPACE -
 Taste aus-  geloest. Das Aufnehm
en eines      neuen Teils geschieh
t automa-     tisch.
              Die Anzeigen: Punkte
(oben links)     und Highscore(
oben rechts)Bitte geben Sie nun
den ge-       wuenschten Schwierig
keitsgrad     ein!(1=leicht,2=mitt
el,3=schwer)
                              VIEL
  SPASS !"
6000 RETURN
7998 STOP
7999 REM Spielende
8000 FOR i=1 TO 50: NEXT i: PRIN
T AT 20,car+1; INK 7; PAPER 1;"I
J": BEEP 1,-50: BEEP 1,-40: PRIN
T AT 20,car+1; PAPER 1;"  "
8100 INK 0: PAPER 7: PRINT AT 8,
0;"Opa Sinclair ist leider mit
    seinem Auto ertrunken !
    "
8200 INPUT "nochmal ? (J/N) ";n$
8300 IF n$="n" OR n$="N" THEN ST
OP
8310 IF pu>=hi THEN LET hi=pu
8400 GO TO 11
8999 STOP
9000 REM Initialisierung
9005 RESTORE 9000
9010 FOR a=USR "a" TO USR "j"+7
9020 READ b: POKE a,b: NEXT a
9030 DATA 128,192,224,240,248,25
2,254,255
9040 DATA 1,3,7,15,31,63,127,255
9050 DATA 255,255,136,85,34,255,
255,255
9060 DATA 255,0,141,255,159,15,4
,31
9070 DATA 255,128,240,200,252,24
8,16,252
9080 DATA 3,3,1,255,255,195,219,
24
9090 DATA 64,160,32,252,255,195,
218,24
9100 DATA 32,32,32,32,32,32,32,3
2
9110 DATA 129,64,0,18,0,72,0,36
9120 DATA 0,1,36,0,72,0,40,2
9900 RETURN
9910 REM Mit GO TO 9920 kann die
     Zahl der verbrauchten Bytes
     festgestellt werden!
9920 PRINT PEEK 23637+256*PEEK 2
3628-23755
9930 STOP
9998 REM Mit GO TO 9999 kann das
     Programm abgesaved werden !
9999 SAVE "Opa Sincl." LINE 1: B
EEP 1,1: VERIFY ""
```

Disk-Master

Welcher Diskettenbenutzer kennt nicht die ewige Sucherei nach einem bestimmten Programm, welches auf einer der vielen Disketten gespeichert ist; dieses Programm schafft Ordnung (TI-99/4A mit Extended-BASIC und Diskettenlaufwerk)

Disk-Master arbeitet ähnlich einem Interpreter, das heißt, Sie geben über die Tastatur wie bei BASIC die Befehle ein, und Disk-Master wird diese Befehle ausführen. Ferner ist Disk-Master mit einer großen Anzahl von Fehlermeldungen ausgestattet, welche die meisten der vorkommenden Fehler abdecken. Doch nun zum Programm:

Initialisieren Sie eine Diskette mittels des Disk-Managers auf den Namen Master. Achten Sie jedoch darauf, daß Sie die Diskette nicht mit einem Kopierschutz versehen, da sonst die Update-Dateien nicht ausgelesen werden können. Tippen Sie nun Disk-Master ein, und speichern Sie das Programm unter dem Namen LOAD auf die Diskette MASTER

Scan: S. Höltgen

ab. Dies hat den Vorteil, daß das Programm bei eingelegter Diskette sofort nach der Auswahl X-BASIC eingelesen und gestartet wird.

Starten Sie jetzt das Programm. Beim ersten Programmstart wird Ihr Laufwerk für circa zwei Minuten in Aktion treten, um die nicht vorhandenen Dateien zu erstellen und zu initialisieren. Bei den weiteren Programmläufen entfällt diese Initialisierungsphase, und das Laufwerk wird nur kurz anlaufen, um den Nullsektor zwecks Identifizierung zu lesen. Das Programm meldet sich jetzt mit

DISK MASTER
* * READY * *
?

und ist bereit, Ihre Befehle zu empfangen.

Disk-Master kennt sechs verschiedene Befehle, die teilweise in Verbindung mit einem oder mehreren Parametern eingesetzt werden. Die Befehle werden im folgenden erklärt.

ERFASSE

Mittels dieses Befehles wird eine Diskette erfaßt. Der Befehl liest von einer bestimmten Diskette den Katalog und erfaßt diesen in einer Datei, um später die Diskette zu einer zugehörigen Datei beziehungsweise einem zugehörigen Programm zu finden.

Tippen Sie ein
ERFASSE (ENTER)

Sie werden jetzt aufgefordert, die zu erfassende Diskette in Laufwerk 1 einzulegen. Danach drücken Sie (ENTER), und der Diskettenkatalog wird ausgelesen. Es können bis zu 40 Eintragungen aus dem Katalog übernommen werden.

Jetzt legen Sie nach der Aufforderung die Master-Disk in Laufwerk 1 und drücken wieder (ENTER).

Der Rechner trägt nun den Diskettennamen ein, sofern die Diskette nicht bereits erfaßt wurde. Danach werden die Programmnamen alphabetisch in das Dateiregister einsortiert. Es wird hierzu der Bubble-Sort benutzt, welcher das Dateiregister direkt auf der Diskette sortiert. Dieser Sort ist für den hier vorliegenden Fall am effizientesten. Es kann jedoch einige Minuten dauern, bis der Vorgang abgeschlossen ist.

Danach wird noch das Alphabetregister reorganisiert. Wenn alle diese Vorgänge abgeschlossen sind, meldet sich Disk-Master mit
* * READY * *
zurück.

LOESCHE Diskettenname

Dieser Befehl löscht aus der Kartei den Diskettennamen und alle zugehörigen Dateinamen. Ferner wird das Alphabetregister reorganisiert.

Der Befehl hat die Form
LOESCHE Diskettenname
wobei für Diskettenname der Name der zu löschenden Diskette einzusetzen ist.

SUCHE Datei

Der Befehl wird benutzt, um die Diskette zu finden, auf welcher der entsprechende Dateiname gespeichert ist.

Sie wollen zum Beispiel wissen, auf welcher Diskette das Programm TEST gespeichert ist und geben ein:
SUCHE TEST

Der Rechner wird Ihnen jetzt die Diskette suchen, auf welcher die Datei gespeichert ist oder, falls die Datei nicht existiert, nichts tun, außer sich mit
* * READY * *
zurückzumelden.

Ist die Datei TEST zum Beispiel auf der Diskette Disk gespeichert, so wird der Rechner Ihnen die Meldung „Datei ist auf Diskette DISK gespeichert" ausgeben.

KATALOG Art (.Gerät)

Dieser Befehl hat, je nach Parameter, verschiedene Funktionen. Der Parameter Art gibt an, was katalogisiert werden soll. Hierin bedeuten
DSK1, DSK2, DSK3
Die Diskette in Laufwerk 1, 2 oder 3 wird katalogisiert.
DISK
Das Gesamtverzeichnis der gespeicherten Disketten wird ausgegeben.
DATEI
Das Gesamtverzeichnis der gespeicherten Dateien wird ausgegeben.

Die Option .Gerät gibt an, auf welchem Ausgabegerät der Katalog ausgegeben werden soll. Wird die Option weggelassen, so wird der Katalog auf dem Bildschirm ausgegeben. Beispiele:

KATALOG DSK1 gibt den Katalog der Diskette in Laufwerk 1 auf dem Bildschirm aus.

KATALOG DATEI.PIO gibt das Gesamtverzeichnis der gespeicherten Dateien auf einem Drucker an der parallelen Schnittstelle aus.

KATALOG DISK.RS232 gibt das Gesamtverzeichnis der gespeicherten Disketten auf der seriellen Schnittstelle aus.

PLATZ

Dieser Befehl zeigt am Bildschirm an, wie viele Disketten und Dateien noch gespeichert werden können. Disk-Master kann bis zu 60 Disketten und 255 Dateien verwalten.

BASIC

Mittels dieses Befehles verlassen Sie Disk-Master und geben die Kontrolle an den BASIC-Interpreter weiter. Tippen Sie BASIC (ENTER) ein, und Ihr Extended-BASIC-Modul meldet sich wieder mit
* READY *

Anmerkungen:

Alle Befehle müssen in Großbuchstaben eingegeben werden. Es empfiehlt sich deshalb, die (ALPHA LOCK)-Taste einzurasten. Bei sämtlichen KATALOG- und SUCHE-Anweisungen kann die Ausgabe durch Drücken einer beliebigen Taste unterbrochen werden. Durch erneutes Drücken einer Taste wird die Ausgabe fortgesetzt. Ein gleicher Dateiname kann auf verschiedenen Disketten vorkommen, muß jedoch mit einem Großbuchstaben beginnen, da es sonst beim Suchen zu einer Fehlermeldung kommt.

Gleiche Diskettennamen dürfen nicht verwendet werden, da sonst ein Auffinden der einzelnen Dateien nicht möglich wäre. Wenn eine Diskette gleichen Namens bereits erfaßt wurde, so wird die Meldung
„Diskette bereits erfaßt"
ausgegeben, und Disk-Master meldet sich mit
* * READY * *
zurück. In diesem Falle kann man den Diskettennamen mittels DISK-MANAGER-Modul jedoch ändern und die Diskette dann erfassen.

Volker Becker

```
1 !##########################
2 !#      DISK - MASTER     #
3 !#  DISKETTENVERWALTUNG   #
4 !#------------------------#
5 !#  TI 99/4 A + X-BASIC   #
6 !#------------------------#
7 !# (c)1984 Volker Becker  #
8 !##########################
9 !
10 OPTION BASE 1
11 DIM STM$(6),KAT$(40),TYP(40),LNG(40),
STP(40),TYP$(5),KAV(5)
12 DATA KATALOG,SUCHE,ERFASSE,LOESCHE,PL
ATZ,BASIC,DIS/FIX,DIS/VAR,INT/FIX,INT/VA
R,PROGRAM,DSK1,DSK2,DSK3,DISK,DATEI
13 CALL DISKINIT
14 OPEN #1:"DSK1.",INPUT ,RELATIVE,INTER
NAL
15 INPUT #1:MAIN$ :: CLOSE #1
16 RESTORE
17 FOR I=1 TO 6 :: READ STM$(I):: NEXT I
18 FOR I=1 TO 5 :: READ TYP$(I):: NEXT I
19 FOR I=1 TO 5 :: READ KAV$(I):: NEXT I
20 CALL CLEAR :: CALL SCREEN(4):: PRINT
"DISK MASTER"
21 FOR I=1 TO 40 :: KAT$(I)="" :: NEXT I
22 PRINT "** READY **"
23 INPUT BEFEHL$
24 CMD$=""
25 FOR I=1 TO LEN(BEFEHL$)
26 IF SEG$(BEFEHL$,I,1)=" " THEN 29
27 CMD$=CMD$&SEG$(BEFEHL$,I,1)
28 NEXT I
29 FOR I=1 TO 6 :: IF CMD$=STM$(I)THEN 3
4
30 NEXT I
31 PRINT "* Befehl unbekannt"
32 CALL SOUND(250,220,0)
33 GOTO 23
34 ON I GOTO 35,40,63,73,94,107
35 FOR I=1 TO LEN(BEFEHL$):: IF SEG$(BEF
EHL$,I,1)=" " THEN 37
```

Scan: S. Höltgen


```
36 NEXT I :: GOTO 108
37 I=I+1 :: CMD$="" :: FOR J=I TO LEN(BE
FEHL$):: IF SEG$(BEFEHL$,J,1)="." THEN 1
14
38 CMD$=CMD$&SEG$(BEFEHL$,J,1):: NEXT J
39 F=0 :: GOTO 119
40 FOR I=1 TO LEN(BEFEHL$)
41 IF SEG$(BEFEHL$,I,1)=" " THEN 44
42 NEXT I
43 GOTO 108
44 I=I+1 :: IF LEN(BEFEHL$)-I>9 THEN 112
45 PROG$=""
46 FOR J=I TO LEN(BEFEHL$)
47 PROG$=PROG$&SEG$(BEFEHL$,J,1)
48 NEXT J
49 OPEN #1:"DSK1.",INPUT ,RELATIVE,INTER
NAL
50 INPUT #1:A$
51 CLOSE #1
52 IF A$<>MAIN$ THEN 53 ELSE 55
53 PRINT "Masterdiskette laden. DanachEN
TER druecken"
54 CALL KEY(0,T,S):: IF S=1 THEN 49 ELSE
 54
55 OPEN #1:"DSK1.DISKFILE",UPDATE,RELATI
VE 60,INTERNAL,FIXED 64
56 OPEN #2:"DSK1.PROGFILE",UPDATE,RELATI
VE 255,INTERNAL,FIXED 64
57 OPEN #3:"DSK1.ALFINIT",UPDATE,RELATIV
E 26,INTERNAL,FIXED 16
58 INPUT #3,REC(ASC(PROG$)-64):NR
59 INPUT #2,REC 0:ENDE
60 IF NR=0 THEN 221
61 INPUT #2,REC NR:NAME$,TYPE,LANG,STYP,
STYP$,SATZ
62 GOTO 208
63 PRINT :: PRINT "Zu erfassende Diskett
e in   Laufwerk 1 laden.          Danac
h ENTER druecken."
64 CALL KEY(0,T,S):: IF S=1 THEN 65 ELSE
 64
65 OPEN #1:"DSK1.",INPUT ,RELATIVE,INTER
NAL
66 INPUT #1:A$,J,J,K
67 IF A$=MAIN$ THEN 263
68 PRINT :: PRINT "Diskettenname : ";A$
:: PRINT :: PRINT "Diskette erfassen"
69 FOR I=1 TO 40 :: INPUT #1:KAT$(I),TYP
(I),LNG(I),STP(I)
70 IF LEN(KAT$(I))=0 THEN 72
71 NEXT I
72 GOTO 182
73 FOR I=1 TO LEN(BEFEHL$)
74 IF SEG$(BEFEHL$,I,1)=" " THEN 77
75 NEXT I
76 GOTO 108
77 CMD$=""
78 FOR J=I+1 TO LEN(BEFEHL$):: CMD$=CMD$
&SEG$(BEFEHL$,J,1):: NEXT J
79 IF LEN(CMD$)>10 THEN 113
80 OPEN #1:"DSK1.",INPUT ,RELATIVE,INTER
NAL
81 INPUT #1:A$
82 CLOSE #1
83 IF A$<>MAIN$ THEN 84 ELSE 86
84 PRINT "Masterdiskette laden. DanachEN
TER druecken"
85 CALL KEY(0,T,S):: IF S=1 THEN 80 ELSE
 85
86 OPEN #1:"DSK1.DISKFILE",UPDATE,RELATI
VE 60,INTERNAL,FIXED 64
87 OPEN #2:"DSK1.PROGFILE",UPDATE,RELATI
VE 255,INTERNAL,FIXED 64
88 OPEN #3:"DSK1.ALFINIT",UPDATE,RELATIV
E 26,INTERNAL,FIXED 16
89 FOR I=1 TO 60 :: INPUT #1,REC I:NAME$
90 IF NAME$=CMD$ THEN 246
91 NEXT I
92 CLOSE #1 :: CLOSE #2 :: CLOSE #3 :: P
RINT :: PRINT "* Diskette ist nicht erfa
sst"
93 GOTO 32
94 OPEN #1:"DSK1.",INPUT ,RELATIVE,INTER
NAL
95 INPUT #1:A$
96 CLOSE #1
97 IF A$<>MAIN$ THEN 98 ELSE 100
98 PRINT :: PRINT "Masterdiskette laden.
 DanachENTER druecken."
99 CALL KEY(0,T,S):: IF S=1 THEN 94 ELSE
 99
100 OPEN #1:"DSK1.DISKFILE",UPDATE,RELAT
IVE 60,INTERNAL,FIXED 64
101 INPUT #1,REC 0:PLZ
102 CLOSE #1 :: PRINT :: PRINT 60-PLZ;"
Disketteneintraege frei"
103 OPEN #1:"DSK1.PROGFILE",UPDATE,RELAT
IVE 252,INTERNAL,FIXED 64
104 INPUT #1,REC 0:PLZ
105 CLOSE #1 :: PRINT :: PRINT 255-PLZ;"
 Dateieintraege frei"
106 PRINT :: GOTO 22
107 CALL CLEAR :: STOP :: END
108 PRINT "* Fehler im Parameter" :: GOT
O 32
109 ON ERROR STOP
110 CALL ERR(A,B)
111 PRINT "* Geraet nicht verfuegbar" ::
 GOTO 32
112 PRINT "* Dateiname zu lang" :: GOTO
32
113 PRINT "* Diskettenname zu lang" :: G
OTO 32
114 GERAET$="" :: FOR I=J+1 TO LEN(BEFEH
L$):: GERAET$=GERAET$&SEG$(BEFEHL$,I,1):
: NEXT I
115 ON ERROR 109
116 F=2
117 OPEN #2:GERAET$,OUTPUT
118 ON ERROR STOP
119 FOR J=1 TO 5 :: IF CMD$=KAV$(J)THEN
121
120 NEXT J :: GOTO 108
121 ON J GOTO 122,122,122,152,152
122 PRINT
123 ON ERROR 109
124 OPEN #1:CMD$&".",INPUT ,RELATIVE,INT
ERNAL
125 ON ERROR STOP
126 PRINT #F:"Diskette katalogisieren"
127 PRINT #F:"-------------------------"
128 PRINT #F
129 INPUT #1:A$,J,J,K
130 PRINT #F:"Diskettenname    :  ";A$ ::
PRINT #F:"Freie Sektoren   :";K :: PRINT
#F:"Belegte Sektoren:";J-K :: PRINT #F
131 PRINT #F:"    Name     Laenge    Typ
S"
132 PRINT #F:"---------- ---- ----------
-"
133 CALL KEY(0,T,S):: IF S=1 THEN 134 EL
SE 135
134 CALL KEY(0,T,S):: IF S=1 THEN 135 EL
SE 134
135 INPUT #1:A$,I,J,K :: IF LEN(A$)=0 TH
EN 146
136 FOR U=LEN(A$)TO 11 :: A$=A$&" " :: N
EXT U
137 A$=A$&STR$(J)
138 FOR U=LEN(A$)TO 15 :: A$=A$&" " :: N
EXT U
139 A$=A$&TYP$(ABS(I))
140 IF ABS(I)=5 THEN 142
141 A$=A$&STR$(K)
142 FOR U=LEN(A$)TO 26 :: A$=A$&" " :: N
EXT U
143 IF I<0 THEN A$=A$&"J"
144 PRINT #F:A$
145 GOTO 133
146 CLOSE #1
147 PRINT #F :: IF F=0 THEN 22
148 CLOSE #2
149 GOTO 22
150 PRINT :: PRINT "Masterdiskette in La
ufwerk  1 laden. dann ENTER druecken"
151 CALL KEY(0,T,S):: IF S=1 THEN 152 EL
SE 151
152 OPEN #1:"DSK1.",INPUT ,RELATIVE,INTE
RNAL
153 INPUT #1:A$
154 CLOSE #1
155 IF A$<>MAIN$ THEN 150
156 ON J-3 GOTO 157,224
157 OPEN #1:"DSK1.DISKFILE",UPDATE,RELAT
IVE 60,INTERNAL,FIXED 64
158 PRINT
```

Scan: S. Höltgen


```
159 Z=0
160 PRINT #F:"Gesamtverzeichnis Diskette
n"
161 PRINT #F:"--------------------------
-" :: PRINT #F
162 PRINT #F:"Nr     Name     Frei Belegt
Sp"
163 PRINT #F:"-- ---------- ---- ------
--"
164 FOR Z=1 TO 60
165 CALL KEY(0,T,S):: IF S=1 THEN 166 EL
SE 167
166 CALL KEY(0,T,S):: IF S=1 THEN 167 EL
SE 166
167 INPUT #1,REC Z:A$,FR,BEL,TR
168 IF LEN(A$)=0 THEN 177
169 B$=STR$(Z)
170 FOR U=LEN(B$)TO 2 :: B$=B$&" " :: NE
XT U
171 A$=B$&A$
172 FOR U=LEN(A$)TO 14 :: A$=A$&" " :: N
EXT U
173 A$=A$&STR$(FR):: FOR U=LEN(A$)TO 20
:: A$=A$&" " :: NEXT U
174 A$=A$&STR$(BEL):: FOR U=LEN(A$)TO 25
:: A$=A$&" " :: NEXT U
175 A$=A$&STR$(INT(TR))
176 PRINT #F:A$
177 NEXT Z
178 PRINT
179 CLOSE #1
180 IF F=0 THEN 22
181 GOTO 148
182 CLOSE #1
183 PRINT :: PRINT "Masterdiskette in La
ufwerk 1laden. Danach ENTER druecken"
184 CALL KEY(0,T,S):: IF S=1 THEN 185 EL
SE 184
185 OPEN #1:"DSK1.",INPUT ,RELATIVE,INTE
RNAL
186 INPUT #1:B$ :: CLOSE #1
187 IF B$<>MAIN$ THEN 183
188 PRINT :: PRINT "Diskette eintragen"
189 OPEN #1:"DSK1.DISKFILE",UPDATE,RELAT
IVE 60,INTERNAL,FIXED 64
190 OPEN #2:"DSK1.PROGFILE",UPDATE,RELAT
IVE 255,INTERNAL,FIXED 64
191 OPEN #3:"DSK1.ALFINIT",UPDATE,RELATI
VE 26,INTERNAL,FIXED 16
192 FOR I=1 TO 60 :: INPUT #1,REC I:V$ :
: IF V$=A$ THEN 265
193 NEXT I
194 FOR I=1 TO 60 :: INPUT #1,REC I:V$ :
: IF V$="" THEN 197
195 NEXT I
196 CLOSE #1 :: CLOSE #2 :: CLOSE #3 ::
PRINT :: PRINT "* Diskettendatei voll" :
: GOTO 32
197 PRINT #1,REC I:A$,K,J-K,(J+2)/9
198 INPUT #1,REC 0:ANZ :: ANZ=ANZ+1 :: P
RINT #1,REC 0:ANZ
199 CLOSE #1
200 DISKNR=I
201 INPUT #2,REC 0:LASTREC
202 PRINT :: PRINT "Sortieren"
203 FOR I=1 TO 40 :: IF LEN(KAT$(I))=0 T
HEN 206
204 CALL BUBBLE(LASTREC,KAT$(I),TYP(I),L
NG(I),STP(I),TYP$(),DISKNR)
205 NEXT I
206 CALL ALFASORT
207 CLOSE #2 :: CLOSE #3 :: GOTO 21
208 IF ASC(PROG$)<>ASC(NAME$)THEN 218
209 IF PROG$=NAME$ THEN 212
210 NR=NR+1
211 IF NR>ENDE THEN 218 ELSE 60
212 INPUT #1,REC SATZ:DISKNAME$
213 PRINT :: PRINT "Datei ist auf Disket
te    ";DISKNAME$;" gespeichert"
214 NR=NR+1
215 IF NR>ENDE THEN 218
216 CALL KEY(0,T,S):: IF S=1 THEN 217 EL
SE 60
217 CALL KEY(0,T,S):: IF S=1 THEN 60 ELS
E 217
218 CLOSE #1 :: CLOSE #2 :: CLOSE #3
219 PRINT
220 GOTO 22
221 CLOSE #1 :: CLOSE #2 :: CLOSE #3
222 PRINT "* Unter diesem Buchstaben
 existiert keine Eintragung"
223 GOTO 32
224 OPEN #1:"DSK1.PROGFILE",UPDATE,RELAT
IVE 255,INTERNAL,FIXED 64
225 PRINT :: PRINT #F:"Gesamtverzeichnis
 Dateien" :: PRINT #F:"------------------
--------" :: PRINT #F
226 PRINT #F:"    Name    Laenge    Typ
S"
227 PRINT #F:"---------- ---- ----------
-"
228 INPUT #1,REC 0:LASTREC
229 FOR I=1 TO LASTREC :: INPUT #1,REC I
:NAME$,TYPE,LANG,STYP,STYP$
230 CALL KEY(0,T,S):: IF S=1 THEN 231 EL
SE 232
231 CALL KEY(0,T,S):: IF S=1 THEN 232 EL
SE 231
232 IF LEN(NAME$)=0 THEN 242
233 FOR U=LEN(NAME$)TO 11 :: NAME$=NAME$
&" " :: NEXT U
234 NAME$=NAME$&STR$(LANG):: FOR U=LEN(N
AME$)TO 15 :: NAME$=NAME$&" " :: NEXT U
235 NAME$=NAME$&STYP$
236 IF ABS(TYPE)=5 THEN 238
237 NAME$=NAME$&STR$(STYP)
238 FOR U=LEN(NAME$)TO 26 :: NAME$=NAME$
&" " :: NEXT U
239 IF TYPE<0 THEN NAME$=NAME$&"J"
240 PRINT #F:NAME$
241 NEXT I
242 PRINT :: CLOSE #1
243 IF F=0 THEN 22
244 CLOSE #2
245 GOTO 22
246 INDEX=I
247 PRINT :: PRINT "Diskettenname loesch
en"
248 PRINT #1,REC INDEX:"",0,0,0
249 INPUT #1,REC 0:NR :: PRINT #1,REC 0:
NR-1
250 INPUT #2,REC 0:LASTREC
251 BACK=0
252 PRINT :: PRINT "Dateinamen loeschen"
253 FOR I=1 TO LASTREC
254 INPUT #2,REC I:NAME$,TYPE,LANG,STYP,
STYP$,DISKNR
255 IF DISKNR=INDEX THEN BACK=BACK+1 ::
GOTO 257
256 PRINT #2,REC(I-BACK):NAME$,TYPE,LANG
,STYP,STYP$,DISKNR
257 NEXT I
258 PRINT #2,REC 0:LASTREC-BACK
259 FOR I=LASTREC-BACK+1 TO LASTREC :: P
RINT #2,REC I:"",0,0,0,0
260 NEXT I
261 CALL ALFASORT
262 CLOSE #1 :: CLOSE #2 :: CLOSE #3 ::
PRINT :: GOTO 22
263 CLOSE #1
264 GOTO 63
265 CLOSE #1 :: CLOSE #2 :: CLOSE #3
266 PRINT :"* Diskette bereits erfasst"
267 GOTO 32
268 SUB DISKINIT
269 OPEN #1:"DSK1.ALFINIT",UPDATE,RELATI
VE 26,INTERNAL,FIXED 16
270 ON ERROR 275
271 INPUT #1,REC 0:TEST
272 ON ERROR STOP
273 CLOSE #1
274 SUBEXIT
275 ON ERROR 277
276 CLOSE #1
277 OPEN #1:"DSK1.ALFINIT",UPDATE,RELATI
VE 26,INTERNAL,FIXED 16
278 FOR I=0 TO 26 :: PRINT #1,REC I:0 ::
 NEXT I :: CLOSE #1
279 OPEN #1:"DSK1.DISKFILE",UPDATE,RELAT
IVE 60,INTERNAL,FIXED 64
280 PRINT #1,REC 0:0
281 FOR I=1 TO 60 :: PRINT #1,REC I:"",0
,0,0 :: NEXT I
282 CLOSE #1
283 OPEN #1:"DSK1.PROGFILE",UPDATE,RELAT
IVE 255,INTERNAL,FIXED 64
```

Scan: S. Höltgen


```
284 PRINT #1,REC 0:0
285 FOR I=1 TO 255 :: PRINT #1,REC I:" "
,0,0,0,0 :: NEXT I
286 CLOSE #1
287 SUBEND
288 SUB ALFASORT
289 PRINT :: PRINT "Reorganisieren"
290 ASCII=64
291 INPUT #2,REC 0:LAUF
292 FOR I=1 TO LAUF :: INPUT #2,REC I:P$
 :: IF LEN(P$)=0 THEN SUBEXIT
293 ASCII1=ASC(P$):: IF ASCII1>ASCII THE
N 296
294 NEXT I
295 SUBEXIT
296 ASCII=ASCII1 :: PRINT #3,REC ASCII-6
4:I
297 GOTO 294
298 SUBEND
299 SUB BUBBLE(LASTREC,KAT$,TYP,LNG,STP,
TYP$(),DISKNR)
300 IF LASTREC=0 THEN I=1 :: GOTO 310
301 FOR I=LASTREC+1 TO 2 STEP -1
302 INPUT #2,REC I-1:KAT1$,TYP1,LNG1,STP
1,TYP1$,DISK1
303 IF KAT1$>=KAT$ THEN 307
304 PRINT #2,REC I-1:KAT1$,TYP1,LNG1,STP
1,TYP1$,DISK1
305 GOTO 310
306 NEXT I
307 PRINT #2,REC I:KAT1$,TYP1,LNG1,STP1,
TYP1$,DISK1
308 IF I=2 THEN 313
309 GOTO 306
310 PRINT #2,REC I:KAT$,TYP,LNG,STP,TYP$
(ABS(TYP)),DISKNR
311 LASTREC=LASTREC+1 :: PRINT #2,REC 0:
LASTREC
312 SUBEXIT
313 PRINT #2,REC 1:KAT$,TYP,LNG,STP,TYP$
(ABS(TYP)),DISKNR
314 LASTREC=LASTREC+1 :: PRINT #2,REC 0:
LASTREC
315 SUBEND
```

Lost in the Antarctica

Im Jahre 1999 scheitert eine der bisher aufwendigsten Antarktis-expeditionen. Schwere Unwetter machen es den fünf Forschern unmöglich, vorwärts zu kommen. Das notdürftig errichtete Lager ist schnell eingeschneit (für alle Atari-Home-Computer)

5. Tag: Die Vorräte beginnen zur Neige zu gehen. Die Außentemperaturen steigen zwar um einige Grad, doch die Windstärke nimmt fast unaufhörlich zu...

8. Tag: Die Stürme lassen nur wenig nach. Die Vorräte reichen noch für höchstens drei Tage. Zwei Expeditionsteilnehmer sind erkrankt...

9. Tag: Es gelingt, einen Hilferuf per Funk zu schicken. Eine Rettungsstation will versuchen, Proviantpakete abzuwerfen...

Drei Stunden später kann das Spezialfahrzeug der For-

Zeilen	Kommentar
1–7	Programmkopf
9	Sprung zum Titel
10	Variablen löschen; Tongeneratoren ausschalten
20	Sprung zum Playfield-Offset
30	TRAP einschalten
40	Sprung zum Main Loop
CAR MOVEMENT	
60	Joystick-Abfrage
70	Gangwahl
80–90	Bewegung des Fahrzeugs und Soundeffekt
95	START – Funktion
100	Rücksprung
MAIN LOOP	
120–160	Jeweilige Flughöhe einnehmen; Sound
170–210	Bewegen der Flugzeuge
220	Ermitteln der Abwurfposition
230–270	Abwurf
280	Nicht gefangen: Sprung nach 330
290–320	Soundeffekt und Display „Gefangen"
330	Soundeffekt und Display „Nicht gefangen"
PLAYFIELD OFFSET	
360	GRAPHICS 7 einschalten; alle Farbregister auf null
370–440	Eisberge zeichnen
450–460	Display anlegen
470–580	PM – Graphics anlegen und Player zeichnen
590–610	Maschinenprogramm „Player nach unten bewegen" in DOWNCODE (String) ablegen
620	Einfärben, Rücksprung
END OF GAME	
635	Tongeneratoren ausschalten
650	Weniger als 10 Pakete gefangen: Sprung nach 760
660–750	Soundeffekt und Display „Gewonnen"
760–800	Soundeffekt und Display „Verloren"
TITEL	
810–1130	Titelbild zeichnen; Musik abspielen
1140–1200	Musik – Datas
FEHLERBEHANDLUNG	
2000	Auto zu weit links
2001	Auto zu weit rechts
2002	TRAP-Info erneuern

Programmstruktur

Scan: S. Höltgen

»Auch wir sind 1985 wieder auf der Hobby-Elektronik!«

Diese übereinstimmende Aussage führender Hersteller beweist die zunehmende Bedeutung dieser Messe. Die Hobby-Elektronik 85 ist der Markt- und Treffpunkt für alle, die sich in ihrer Freizeit mit einem anspruchsvollen Hobby beschäftigen. Sie ist die zentrale Herbstveranstaltung auf den Gebieten praktische Elektronik, Mikrocomputer, Modellbau und Modelleisenbahnen und für viele Bastler und Tüftler ein Pflichttermin. Wenn Sie also den Anschluß nicht verlieren wollen, nützen Sie die Hobby-Elektronik 85 und stellen Sie aus!

Ausführliche Informationen erhalten Sie von der Stuttgarter Messe- und Kongreß-GmbH, Postfach 990, D-7000 Stuttgart 1, Telefon (07 11) 25 89-1, Telex 7 22 584 killb d.

HOBBY-ELEKTRONIK 85

2

RADIX

Achtung!!!

RADIX Bürotechnik
Rappstraße 13 · 2000 Hamburg 13
Tel. 040/441695 · Telex 213682 radix d
tägl. 10.00-12.30 + 13.30-18.30 Uhr
Sa. 10.00-13.00 Uhr
Verkaufsstelle Kiel: Ziegelteich 23 · 2300 Kiel 1

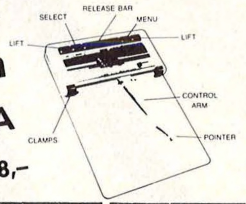
SOFTWARE

- Editor Assembler **189,–**
- Extened Basic - orig. **295,–**
- dito - Nachbau **248,–**
- Tunnels of Doom **79,–**
- Adventure Modul **79,–**
- TI-Logo dt **320,–**
- Defender 69,-
- Der Schwarze Kristall **49,–**
- Basic-Campiler 198,-
- Flugsimulation **49,–**
- Textverarbeitung **79,–**
- Terminal Emulator **99,–**
- USCD-Pascal Komplett 1098,-

MSX-Computer
Philips MSX 8010 mit Kassettenrecorder 799,-
SPECTRA Video incl. 1 Kassettenspiel Ihrer Wahl. 998,-

TI 99-HARDWARE

Super Sketch
TI 99/4A
248,–



NEU!

Über die Innereien des TI 99/4A, eines doch recht erfolgreichen Heimcomputers, ist, bedingt wohl auch durch eine sehr zurückhaltende Informationspolitik seitens des Herstellers, nicht sehr viel bekannt. Dieses Buch soll dem interessierten Anwender helfen, hinter die Geheimnisse des Betriebssystems des TI 99/4A zu blicken.

DM 38,–

NEU! TI 99/4A Mini-Assembler

Mit dieser Hardware-Erweiterung können Sie nur mit Ext.-Basic und Kassettenrecorder mit Assembler programmieren. incl. Software

149,–

HARDWARE

- TI-Box, Disklautw., Contr. 32 K-Erw. + 10 Disk. **1898,-**
- Aufrüstung auf DOS 80 für ds/dd-Laufwerke (360 KB) ab **148,-**
- 32 K-Erw. extern **428,-**
- Druckerinterface extern **348,-**
- Drucker GP 50 + Kabel + Druckerinterface extern **798,-**
- dito mit GP 550 **1198,-**
- dito mit EPSON RX 80 **1448,-**
- dito mit EPSON FX 80 **1990,-**
- Sprachsteuereinheit **348,-**
- Akustikcoupler mit FTZ-Nr. dataphon **298,-**

Brother, Silver Reed, Seikosha, HP, 3 M Scotch, Apple, Sanyo, Commodore, Atari

Zubehör:
Monitore, Disketten, Druckerbuffer, Disketten-Organisation, Computer-Möbel, Papier
Abdeckhauben Kunstleder mit Leinen

Fordern Sie bitte unsere speziellen Preislisten der einzelnen Marken - Systeme an. (mit Typenangaben)
Preise Stand 03.85. Alle Preise incl. MwSt. · Preisliste anfordern! · Lieferung erfolgt per NN oder gegen Verrechnungscheck.
Bestellungen über DM 500,- werden frei Haus geliefert, unter DM 500,- werden DM 5,- Versandpauschale berechnet.

ACHTUNG! TI-99/4A Besitzer – ACHTUNG! TI-99/4A Besitzer

Peripherie

orig. TI-Peripherie Erw.-Box mit Disk-Steuerkarte u. Laufwerk	1.480,-
32 K-Byte RAM	425,-
RS 232 Karte	398,-

Externe Erweiterungen

32 K-Byte RAM	299,-
32 K-Byte RAM mit Centronics-Interface	399,-

Zubehör

Graphic Tableau	248,-
Cartridge Expander (für 3 Module)	128,-
Akustikkoppler-Dataphon	298,-
DIN A4-4 Farb.-Drucker/Plotter	898,-
Slim Line Disk-Laufwerk (DS DD)	650,-
Einbausatz für 2 Slim Line Laufwerke in orig. Box	96,-

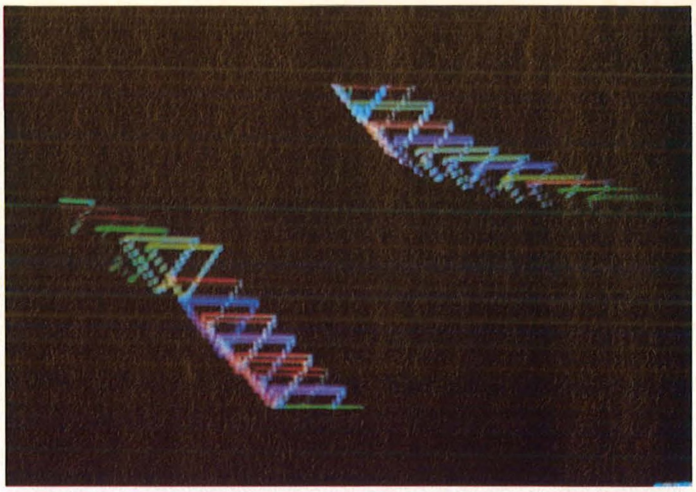
Module

Extended Basic (deutsch)	248,-
Mini Memory	290,-
Terminal Emulator II	85,-
Multiplan	320,-
Editor Assembler	220,-
Spiele von	30,- bis 99,-
z. B. Moonmine	65,-

Alle Preise incl. MwSt. zuzügl. 5,- DM Versandkosten. Lieferung per Nachnahme oder Vorkasse, ab 200,- DM versandkostenfrei.
Fordern Sie unsere kostenlose Preisliste an.

Programm-Service REIS GmbH
5584 Bullay
Bergstraße 80
Telefon 06542/2715

Psychedelic-Dreams



Die 28 Sprites des TI-99/4A mit Extended-BASIC bewegen sich als Ketten unterschiedlicher Form und Länge über den Bildschirm, versammeln sich plötzlich, bleiben stehen und formieren sich neu: ein farbenfrohes Schauspiel

Das Programm startet mit einem Menü, in dem man die Form und den Vergrößerungsfaktor, den die Sprites im Hauptprogramm erhalten sollen, eingeben kann. Danach bewegen sich 28 Sprites in den festgelegten Formen über den Bildschirm. Möchte man deren Geschwindigkeit abändern, so drückt man eine Taste, bis ein kurzes Piepen ertönt. Jetzt ist der Computer zur Eingabe bereit. Entweder ändert man jetzt die Geschwindigkeit mit der Eingabe einer Zahl (0–31) ab, oder man kehrt durch Drücken der ENTER-Taste in das Menü zurück.

Probieren Sie möglichst viele Geschwindigkeiten aus, denn so können viele verschiedene Muster entstehen.

Ralf Gröschel

```
100 GOTO 230
110 !
120 !****************** Schnellstarter **
****************
130 !
140 CALL CHAR :: CALL CLEAR :: CALL DELS
PRITE
150 CALL KEY :: CALL MAGNIFY :: CALL MOT
ION
160 CALL SCREEN :: CALL SPRITE
170 ER,ZEI,SPA,K,S,I,WA,MA,X,Y,GESCHW
180 DIM A$(6):: FRA$
190 !@P-
200 !
210 !****************** Graphikzeichen **
****************
220 !
230 A$(1)="FF80808080808080808080808080808
0FFFF0101010101010101010101010101FF"
240 A$(2)=RPT$("F",64)
250 A$(3)="071830604080808080808040603018
07E0180C06020101010101010102060C18E"
260 A$(4)="071F3F7F7FFFFFFFFFFFFFF7F7F3F1
F07E0F8FCFEFEFFFFFFFFFFFFFFFEFEFCF8E"
270 A$(5)="0101020204040808101020204040
80FF8080404020201010080804040202 01FF"
280 A$(6)="0101030307070F0F1F1F3F3F7F7FF
FFF8080C0C0E0E0F0F0F8F8FCFCFEFEFFFF"
290 !
300 !********************** Menue *******
****************
310 !
320 CALL CLEAR :: CALL SCREEN(8):: CALL
DELSPRITE(ALL)
330 DISPLAY AT(3,10):"M E N U E"
340 DISPLAY AT(4,10):"========="
350 DISPLAY AT(8,3):"QUADRAT............
...1"
360 DISPLAY AT(10,3):"QUADRAT AUSGEMALT.
....2"
370 DISPLAY AT(12,3):"KREIS.............
....3"
380 DISPLAY AT(14,3):"KREIS   AUSGEMALT.
....4"
390 DISPLAY AT(16,3):"DREIECK...........
....5"
400 DISPLAY AT(18,3):"DREIECK AUSGEMALT.
....6"
410 DISPLAY AT(22,10):"IHRE WAHL ?"
420 CALL KEY(0,K,S)
430 IF K<49 OR K>54 THEN 420
440 WA=K-48
450 CALL CLEAR
460 DISPLAY AT(9,11):"WELCHER"
470 DISPLAY AT(12,3):"VERGROESSERUNGSFAK
TOR ?"
480 DISPLAY AT(15,10):"(1 ODER 2)"
490 CALL KEY(0,K,S)
500 IF K<49 OR K>50 THEN 490
510 MA=K-46
520 !
530 !****************** Hauptprogramm ***
****************
540 !
550 CALL CLEAR
560 CALL SCREEN(2)
570 CALL MAGNIFY(MA)
580 GESCHW=5
590 CALL CHAR(100,A$(WA))
600 CALL CLEAR
610 !
620 !***************** Spriteerzeugung **
****************
630 !
640 ZEI=INT(RND*190)+1 :: SPA=INT(RND*23
0)+20
650 FOR I=1 TO 28 :: CALL SPRITE(#I,100,
INT(RND*14)+3,ZEI,SPA):: NEXT I
660 !
670 !****************** neue Richtung ***
****************
680 !
690 X=(INT(RND*8)-4)
700 IF X=0 OR ABS(X)<3 THEN 690
710 Y=(INT(RND*8)-4)
720 RANDOMIZE
730 CALL KEY(0,K,S):: IF S=0 THEN 830
740 !
750 !********** neue Geschwindigkeit eing
eben ***********
760 !
770 ACCEPT BEEP SIZE(2)VALIDATE(DIGIT):F
RA$
780 IF FRA$="" THEN 320 ELSE GESCHW=VAL(
FRA$)
790 IF GESCHW>31 THEN 770
800 !
810 !**************** Richtungsaenderung
****************
820 !
830 FOR I=1 TO 28 :: CALL MOTION(#I,X*GE
SCHW,Y*GESCHW):: NEXT I
840 ER=ER+1 :: IF ER>10 THEN ER=0 :: GOT
O 640
850 GOTO 690
```

Scan: S. Höltgen

BASIC ist nicht gleich BASIC

Noch sind die rosigen Zeiten einheitlicher Software für alle Home-Computer in weiter Ferne. Damit es dem Home-Computer nicht die Sprache verschlägt, hier die wichtigsten Tips zum Konvertieren

Wohl jeder Home-Computer-Besitzer kennt die Situation. Da glaubt man sich am Ziel seiner Träume: Man hat ein Programm gefunden, abgedruckt als Listing in einer Zeitschrift oder einem Buch, nach dem man eigentlich schon lange gesucht hat; doch zu früh gefreut: Es wurde nicht für den Home-Computer geschrieben, den man sein Eigen nennt. Zwar ist die verwendete Programmiersprache BASIC; bekanntlich gibt es jedoch nicht „das BASIC", sondern nur verschiedene BASIC-Dialekte, zwischen denen unterschiedliche Sprachbarrieren aufgerichtet sind. Verloren ist das Programm für den eigenen Rechner also noch nicht, denn diese Dialekte lassen sich ineinander „übersetzen". Bei dieser Art von Übersetzung ersetzt der Programmierer beim Eintippen im einfachsten Fall ein dem Computer fremdes Sprachelement durch ein entsprechendes aus dem eigenen Sprachschatz. Oftmals ist die Sache etwas schwieriger, nämlich dann, wenn der betreffende Dialekt kein äquivalentes Sprachelement zur Verfügung stellt. Es hängt nun von der Höhe der Sprachbarriere ab, mit welchem Aufwand eine Konvertierung (Umschreiben) des Programmes verbunden ist. Da gibt es ziemlich aussichtslose Fälle, an die man sich erst gar nicht heranwagen sollte, um sich unnötigen Ärger zu ersparen. Bei einer ganzen Reihe von Rechnerpaaren jedoch lohnt sich der Versuch, zumindest in der einen Übersetzungsrichtung. Wenn in einem Programm ein oder mehrere Teile in Maschinensprache geschrieben sind, ist die Frage wichtig, mit welchen Mikroprozessoren die beiden Home-Computer arbeiten.

Bei Maschinencode schwierig

Gehören sie zwei verschiedenen Reihen an (was zum Beispiel beim Z80 und beim 6502 der Fall ist), dann sind detaillierte Kenntnisse beider Mikroprozessoren unbedingte Voraussetzung, und auch mit diesen Kenntnissen ist eine erfolgreiche Konvertierung sehr mühsam und zeitaufwendig. Mit gleichen oder sehr ähnlichen Mikroprozessoren auf beiden Home-Computern können jedoch trotzdem einer Konvertierung noch viele Stolpersteine im Weg liegen: Vor allem die Benutzung von gerätespezifischen ROM-Routinen oder Adressen. Um dies beurteilen zu können, benötigt man jedoch ein Assembler-Listing des Maschinenprogramms, welches meistens nicht vorliegt. In der Regel bekommt man nämlich das Maschinenprogramm zunächst nur in sehr unfreundlicher Form zu Gesicht, nämlich „getarnt" in DATA-Zeilen. Nur ein Disassembler könnte da die Lage verbessern.

Wegen der genannten Schwierigkeiten haben reine BASIC-Programme die größten Aussichten, erfolgreich übersetzt zu werden. Allgemein läßt sich sagen, daß eine Konvertierung von einem Home-Computer mit komplexerem Sprachangebot zu einem mit weniger Sprachelementen wahrscheinlich langwieriger ist als die Konvertierung in umgekehrter Richtung, vor allem dann, wenn im Programm viele Finessen des komplexen Dialekts Verwendung finden.

Daneben spielen jedoch noch andere Fragen eine Rolle: Drei der wichtigsten sind: Werden Systemvariablen gepoket? Wie groß ist das Zeilen-Spalten-Raster auf dem Bildschirm? Werden Sprites benutzt? POKE-Befehle mit Änderung von Systemvariablen spielen erfahrungsgemäß bei den Commodore-Rechnern VC20 und C64 eine große Rolle. Es zeigt sich dort, daß man nicht darauf vertrauen kann, daß zwei Home-Computer-Typen, die vom gleichen Hersteller kommen, besonders konvertierungs-

Scan: S. Höltgen

freundlich sind oder gar keine benötigen. Insbesondere bei POKE-Befehlen und ROM-Routinen kommt man zwischen VC20 und C64 nicht ohne Vergleichstabellen aus, in denen den Adressen des einen Rechners die entsprechenden des anderen gegenübergestellt sind.

Auf eine Zeile kommt es nicht an

Die Bildschirmaufteilung schafft dann wenig Probleme, wenn sowohl die Zeilenzahl als auch die Spaltenzahl auf dem Home-Computer, zu dem konvertiert werden soll, mindestens genauso groß sind wie auf dem Home-Computer, für den das Programm ursprünglich geschrieben worden war. Etwas ungewöhnlich wird allerdings bei viel Textausgabe sein, daß der Text sich in der oberen linken Ecke des Bildschirms zusammendrängt. Wen das stört, der muß die Bildschirmaufteilung neu durchdenken. Dies ist in jedem Fall dann erforderlich, wenn weniger Zeilen oder Spalten zur Verfügung stehen und vorher der ganze Bildschirm ausgenutzt wurde. Übersetzungen zum Beispiel vom VC20 (22 Spalten, 23 Zeilen) zum C64 (40 Spalten, 25 Zeilen) oder vom Sinclair-Spectrum (32 Spalten, 22 Zeilen) zum TI-99/4A (32 Spalten, 24 Zeilen) sind also hinsichtlich der Bildschirmaufteilung weniger problematisch als die jeweils umgekehrten Konvertierungsrichtungen. Zu beachten ist, daß manche Home-Computer die Zeilen und Spalten nicht mit 0 beginnend durchzählen, wie die meisten anderen, sondern mit 1 (zum Beispiel der TI-99/4A). Als sehr hilfreich erweist sich die Möglichkeit einiger Home-Computer, verschiedene Modi mit jeweils anderen Zeilen-Spalten-Aufteilungen zu wählen, wie zum Beispiel bei den Atari-Rechnern, den MSX-Rechnern und dem Schneider CPC.

Wenn auf Sprites bei der Übersetzung verzichtet werden muß, so lassen sich diese in BASIC nur schwer und dann wesentlich langsamer simulieren. Neben diesen pauschalen Grundregeln kommt es beim konkreten Beispiel auf sehr viele einzelne Tips an, von denen jetzt die wichtigsten aufgezählt werden sollen. Schon bei der Obergrenze, bis zu der Zeilennummern möglich sind, ergeben sich erhebliche Unterschiede. Während beim Spectrum schon bei 9999 Schluß ist, kann der Schneider als letzte Zeilennummer noch 65535 verkraften. Zum Beispiel vom Schneider zum Spectrum kann es also erforderlich sein, im Programm Zeilennummern zu ändern. Dabei darf nicht vergessen werden, daß die Umnumerierung auch Änderungen in allen Sprungbefehlen erforderlich macht, die ursprünglich auf den Bereich über 9999 zielten. Während REM-Zeilen, in denen sich meist nur Programm-Erläuterungen befinden, einfach ignoriert werden, wenn sie angesprungen werden, nehmen es die meisten Home-Computer übel, wenn ein Sprungbefehl zu einer Zeile zielt, die im Programm gar nicht vorhanden ist. Probleme damit kann es jedoch nur beim Umschreiben von Programmen weniger Home-Computer-Typen (zum Beispiel Spectrum) geben, bei denen ein Sprung zu einer nicht vorhandenen Zeile zu keiner Fehlermeldung führt, sondern zur nächstgrößeren vorhandenen Programmzeile gerückt wird.

LET vor Zuweisungen kann bei den meisten Home-Computern entfallen, einige (zum Beispiel der Spectrum) kommen jedoch nicht ohne aus. Leerzeichen zwischen den einzelnen Elementen einer Programmzeile sind bei einigen nicht (z.B. Commodore), bei anderen (z.B. Schneider) wohl erforderlich. Bedingte Sprünge beinhalten bei den Sinclair-Rechnern immer das Wortpaar THEN GOTO, beim TI-99/4A darf dort nur THEN stehen, den meisten anderen ist es egal, welche von beiden Varianten man wählt. Beim Umschreiben auf den TI-99/4A können logische Verknüpfungen Kopfzerbrechen bereiten, denn er kennt weder AND noch OR, NOT oder XOR. Nur durch geschickte Kombination mehrerer IF-Anweisungen läßt sich dieses Manko ausgleichen. Ausgaben auf den Bildschirm oder einen Drucker sind fast bei jedem Home-Computer anders verwirklicht. Während zum Beispiel beim Commodore zum Erreichen einer bestimmten Bildschirmposition jeweils einzelne Schritte um eine Zeile oder Spalte über Cursorsteuertasten ausgeführt werden müssen, kann zum Beispiel bei den Sinclair-Rechnern hinter PRINT AT

direkt die Zeilen- und die Spaltennummer angegeben werden. Zur Umlenkung einer Ausgabe auf einen Drucker muß bei einigen nur PRINT durch LPRINT oder PRINT durch PRINT#8 ersetzt werden, bei anderen ist zuvor eine Öffnung des Druckerkanals mit OPEN nötig.

Auch Namen ändern sich

Hinsichtlich der Auswahl an Funktionen sind ebenfalls häufig Unterschiede anzutreffen. Die oft verwendete Zufallsfunktion arbeitet bei einem Teil der Home-Computer nur mit einem Parameter in Klammern hinter dem Wort RND, bei anderen genügt RND allein. Die kurzzeitige Abfrage der Tastatur mit INKEY$ muß oftmals in GET mit nachfolgender Stringvariablen oder OPEN#1,4,0,"K:":GET#1,K oder den Aufruf eines Unterprogrammes umgewandelt werden. Auch Variablennamen können oft nicht einfach übernommen werden. Es variieren die maximale Länge eines Namens, die Anzahl der Zeichen, die zur Unterscheidung herangezogen werden und welche Zeichen im Namen erlaubt sind. Um Ton, Farbe oder Grafik auf den anderen Rechner umzusetzen, empfiehlt es sich meistens, die entsprechenden Teile komplett neu zu schreiben, so groß sind die Unterschiede zwischen den Home-Computern. Bei Stringvariablen wird bei den Atari-Rechnern oft vergessen, daß sie in jedem Fall dimensioniert werden müssen. Eine zweimalige Dimensionierung

des gleichen Feldes ist nur bei wenigen Home-Computern möglich. String-Verarbeitung ist ein weiterer Bereich, bei dem unterschiedliche Handhabung berücksichtigt werden muß. Wenn zum Beispiel Anfangs-, End- oder Mittelstücke eines Textes herausgegriffen werden sollen, benutzt fast jeder Home-Computer eine andere Methode.

Um die Anzahl der Änderungen zu reduzieren, sollten Autoren den Konvertierungsaspekt bei der Programmentwicklung berücksichtigen. *br*

Sinclair ZX81, Spectrum	**Commodore VC20, C64**	**Texas Instruments TI-99/4A**	**Atari 600XL, 800XL**	**MSX-Rechner**	**Schneider-CPC**
Zeilennummern					
bis 9999	bis 63 999	bis 32 767	bis 32 767	bis 65 529	bis 65 535
Bildschirmaufteilung (für Texte)					
32 Spalten (0 bis 31) 22 Zeilen (0 bis 21)	VC20 22 Spalten (0 bis 21) 23 Zeilen (0 bis 22) C64 40 Spalten (0 bis 39) 25 Zeilen (0 bis 24)	32 Spalten (1 bis 32) 24 Zeilen (1 bis 24)	40 Spalten (0 bis 39) 24 Zeilen (0 bis 23) oder 20 Spalten (0 bis 19) 24 Zeilen (0 bis 23) oder 20 Spalten (0 bis 19) 12 Zeilen (0 bis 11)	40 Spalten (0 bis 39) 24 Zeilen (0 bis 23) oder 32 Spalten (0 bis 31) 24 Zeilen (0 bis 23)	80 Spalten (0 bis 79) 25 Zeilen (0 bis 24) oder 40 Spalten (0 bis 39) 25 Zeilen (0 bis 24) oder 20 Spalten (0 bis 19) 25 Zeilen (0 bis 24)
Positioniertes Drucken (Beispiel)					
PRINT AT 5, 10; "MUT"	PRINT ,, **SQ** **QQQQ**"; TAB (10); "MUT"	CALL HCHAR (5, 10, 77, 1) CALL HCHAR (5, 11, 85, 1) CALL HCHAR (5, 12, 84, 1)	POS. 10,5: PRINT "MUT"	LOCATE 10,5: PRINT "MUT"	LOCATE 10,5: PRINT "MUT"
Namenslänge von Zahlvariablen					
unbegrenzt	maximal 2 Zeichen	maximal 15 Zeichen	unbegrenzt	unbegrenzt	maximal 40 Zeichen
Zugelassene Namen von Textvariablen					
einzelner Buchstabe gefolgt vom Zeichen $	wie eine Zahl- variable, jedoch maximal drei Zeichen; letztes Zeichen ein $	wie eine Zahl- variable; letztes Zeichen ein $	unbegrenzt; letztes Zeichen ein $	wie eine Zahl- variable; letztes Zeichen ein $	wie eine Zahl- variable; letztes Zeichen ein $
Textbehandlung (Beispiele)					
A$(TO 4) A$(LEN(A$)-2 TO LEN(A$)) A$(6 TO 10) A$(4 TO) A$(7)	= LEFT$(A$,4) = RIGHT$(A$,3) = MID$(A$,6,5) = MID$(A$,4, LEN(A$)-3) = MID$(A$,7,1)	= SEG$(A$,1,4) = SEG$(A$, LEN(A$)-2,3) = SEG$(A$,6,5) = SEG$(A$,4, LEN(A$)-3) = SEG$(A$,7,1)	= A$(1,4) = A$(LEN(A$)- 2,LEN(A$)) = A$(6,10) = A$(4, LEN(A$)) = A$(7,7)	= LEFT$(A$,4) = RIGHT$(A$,3) = MID$(A$,6,5) =vMID$(A$,4, LEN(A$)-3) = MID$(A$,7,1)	= LEFT$(A$,4) = RIGHT$(A$,3) = MID$(A$,6,5) = MID$(A$,4, LEN(A$)-3) = MID$(A$,7,1)
Tastaturabfrage					
INKEY$	GET A$	CALL KEY(0,K,ST)	OPEN#1,4,0, "K:" GET#1,K	INKEY$	INKEY$
Zufallsfunktion					
RND	RND(1)	RND	RND(1)	RND(1)	RND(1)

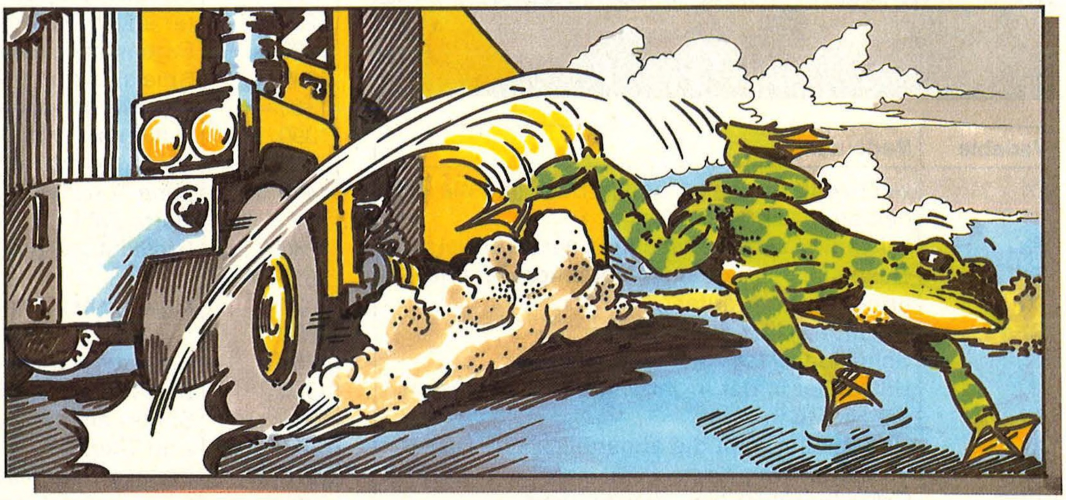
Die wichtigsten Übersetzungshilfen beim Konvertieren von Programmen

Scan: S. Höltgen

Frogger

Auch auf dem TI-99/4A mit Extended-BASIC hat der Frosch des Spielhallenhits noch nicht schwimmen gelernt und muß, nachdem er die gefährliche Straße hinter sich gebracht hat, im Fluß auf hilfreiche Gesellen aufspringen

Der Frosch wird mit den Pfeiltasten in die vier Richtungen bewegt. Gelangt man mit mehr als drei Fröschen in das Loch am Ziel, kommt man automatisch in die nächste Runde. Von Runde zu Runde hat es der Frosch jedoch schwerer, dieses Ziel zu erreichen. Bei der schnellen Reaktion der Tasten kommt es auf das richtige Tastengefühl an. Während des Spiels muß die ALPHA-LOCK-Taste gedrückt sein. *Luis Moreno*



```
100 CALL TITEL
110 CALL DELSPRITE(ALL):: CALL CHAR
    SET
120 F=0 :: L=5 :: R=1
130 CALL CLEAR :: CALL SCREEN(2)
140 CALL MAGNIFY(3)
150 CALL CHAR(96,"01572523211F07030
    31F3B13090408008 0EAA4C484F8E0C0
    E0F8DCC890201000")
160 CALL CHAR(100,"01070503010FB7C3
    070F1F1B112020C080E0A0C080F0EDC
    3E0F0F8D888040403")
170 CALL CHAR(104,"001C1C48BFBEF9B9
    B9F9BEBF481C1C0000383810FF3ECEC
    ECECE3EFF10383800")
180 CALL CHAR(108,"000038FFFF7F44FC
    FC447FFFFF3800000001FFFFE0AFFF
    FFFFF0AFEFF1F0000")
190 CALL CHAR(112,"00000036724BFFE
    FEBF24670300000000001CDCC83BFF7
    A7AFF3BC8DC1C0000")
200 CALL CHAR(116,"00074F78487FBFB9
    B9BF7F48784F070000F8F41211FFFE0
    E0EFEFF1112F4F800")
210 CALL CHAR(124,"000C010306070EAF
    FFAF0F0603010C0000FB0C74FAFCF97
    777F9FCFA7408F300")
220 CALL CHAR(128,"00F6921F3F7EFBEF
    EFFB7E3F1F92F600009CA0E0F0F1BAF
    FFFBAF1F0E0A09C00")
230 CALL CHAR(132,"00000103070F0301
    0000010000000000000F0F8FCFEFEF
    EFCFCFCF")
240 !******* SCHIRM ********
250 !******* AUFBAU ********
260 CALL CHAR(120,"2442C121160E0698
    "):: CALL COLOR(12,12,1)
270 CALL HCHAR(2,1,120,64):: CALL H
    CHAR(2,15,32,3):: CALL HCHAR(3,
    15,32,3):: CALL HCHAR(12,1,120,
    64)
280 CALL CHAR(132,"0000000000000000
    "):: CALL COLOR(13,1,6):: CALL
    HCHAR(4,1,132,256)
290 ! ****** ANZEIGE *******
300 FOR X=3 TO 8 :: CALL COLOR(X,12
    ,1):: NEXT X
310 DISPLAY AT(24,1):"FROGS: 0" ::
    DISPLAY AT(24,11):"LIVES: 5"
320 DISPLAY AT(24,21):"ROUND: 1"
330 !******* AUTOS  ********
340 CALL DELSPRITE(ALL)
350 CALL SPRITE(#2,104,15,153,100,#
    3,108,7,137,158,#4,108,12,153,2
    2)
360 CALL SPRITE(#5,112,8,121,32,#6,
    116,10,105,210,#7,116,11,121,15
    4)
370 CALL MOTION(#2,0,-20-R,#3,0,-22
    -R,#4,0,-20-R)
380 CALL MOTION(#5,0,-24-R,#7,0,-24
    -R,#6,0,-25-R)
390 !******* TURTLE ********
400 CALL SPRITE(#8,124,16,73,49,#9,
    124,16,73,145)
410 CALL SPRITE(#10,124,5,41,49,#11
    ,124,2,41,200)
420 CALL MOTION(#8,0,-16-R,#9,0,-16
    -R)
430 CALL MOTION(#10,0,-16-R,#11,0,-
    16-R)
440 CALL SPRITE(#12,128,5,57,33,#13
    ,128,15,57,129,#14,128,8,57,225
    )
450 CALL MOTION(#12,0,16+R,#13,0,16
    +R,#14,0,16+R)
```

Scan: S. Höltgen

Variable	Bedeutung
F	Frösche, die angekommen sind
L	Anzahl der verbliebenen Leben
R	Rundenzahl
Y,X	Senkrechte/waagerechte Pixelreihe, wo sich der Frosch befindet

Die wichtigsten Variablen

Zeilen	Kommentar
100	Titelbildaufruf
120	Variablen
130–140	Bildschirmfarbe, Spritegröße
150–230	Spritemusterdefinition
240–280	Schirmaufbau
290–320	Anzeigen
330–380	Autos: Sprites und Bewegung
390–470	Wesen: Sprites und Bewegung
480	X/Y-Koordinaten des Frosches
510–590	Bewegung des Frosches während der Straßenüberquerung und Zusammenstoß
600–630	Frosch tot. Variablenneustellung
640–750	Bewegung des Frosches während der Flußüberquerung und Zusammenstoß
760–830	Frosch springt ins Loch
840–870	Spielende; neues Spiel?
880–890	5 Frösche ins Loch gekommen, Melodie ertönt, Beginn der nächsten Runde
900–910	3 oder 4 Frösche ins Loch gekommen; Beginn der nächsten Runde
920–1370	Titelbild (man kann diese Zeilen weglassen; Zeile 100 muß dann auch weggelassen werden)

Programmstruktur

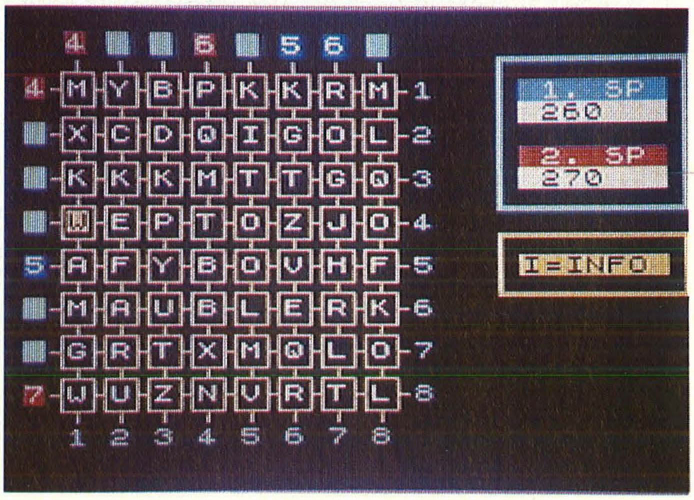
```
460 CALL SPRITE(#15,128,2,25,55,#16
    ,128,15,25,241,#17,128,8,25,177
    )
470 CALL MOTION(#15,0,16+R,#16,0,16
    +R,#17,0,16+R)
480 Y=169 :: X=129
490 !********* FROG ********
500 CALL SPRITE(#1,96,13,Y,X)
510 !******** SPIEL ********
520 !******** UNTEN ********
530 CALL KEY(0,K,S):: IF S=0 THEN C
    ALL COINC(ALL,U):: IF U=-1 THEN
     620 :: GOTO 530
540 A=8*((K=83)-(K=68)):: B=8*((K=6
    9)-(K=88))
550 X=X+A :: Y=Y+B
560 IF X<16 OR X>240 OR Y>177 THEN
    X=X-A :: Y=Y-B :: GOTO 530
570 IF Y<89 THEN 660
580 CALL PATTERN(#1,100):: CALL LOC
    ATE(#1,Y,X)
590 CALL PATTERN(#1,96):: CALL LOCA
    TE(#1,Y,X):: CALL COINC(ALL,U):
    : IF U=-1 THEN 620 :: GOTO 530
600 !******** FROG *********
610 !**** UEBERFAHREN *****
620 CALL SOUND(200,-6,2):: CALL MOT
    ION(#1,0,0):: CALL PATTERN(#1,1
    32):: L=L-1 :: DISPLAY AT(24,18
    )SIZE(1):USING "#":L
630 IF L=0 AND F>2 THEN 900 ELSE IF
     L=0 THEN 840 :: GOTO 480
640 ! ******* SPIEL ********
650 ! ******* OBEN  ********
660 Y=Y+8 :: CALL LOCATE(#1,Y,X)::
    GOTO 680
670 CALL KEY(0,K,S):: IF S=0 THEN 6
    70
680 B=16*((K=69)-(K=88))
690 CALL POSITION(#1,Y,X)
700 Y=Y+B
710 CALL PATTERN(#1,100):: CALL LOC
    ATE(#1,Y,X):: CALL PATTERN(#1,9
    6)
720 IF Y=9 THEN 780
730 CALL COINC(ALL,U):: IF U=0 THEN
     620
740 Q,RR=((Y=73 OR Y=41)-(Y=57 OR Y
    =25))
750 CALL MOTION(#1,0,(16*Q)+(R*RR))
    :: GOTO 670
760 ********FROG**********
770 *****ANGEKOMMEN*******
780 CALL MOTION(#1,0,0):: CALL POSI
    TION(#1,Y,X)
790 FOR H=104 TO 128
800 IF H=X THEN 830
810 NEXT H
820 GOTO 620
830 CALL SOUND(400,1047,3,1319,3,65
    8,4,-8,8):: CALL MOTION(#1,0,0)
    :: F=F+1 :: DISPLAY AT(24,8)SIZ
    E(1):USING ("#"):F :: IF F=5 TH
    EN 880 :: GOTO 480
840 DISPLAY AT(1,13):"GAME OVER"
850 FOR X=1 TO 200 :: NEXT X :: DIS
    PLAY AT(1,10):"NOCHMAL J/N"
860 CALL KEY(0,K,S):: IF S=0 THEN 8
    60
870 IF K=74 THEN 120 ELSE END
880 CALL SIEG :: R=R+1 :: DISPLAY A
    T(24,28)SIZE(1):USING "#":R ::
    L=5 :: F=0
890 DISPLAY AT(24,8)SIZE(1):USING "
    #":F :: DISPLAY AT(24,18)SIZE(1
    ):USING "#":L :: GOTO 340
900 R=R+1 :: DISPLAY AT(24,28)SIZE(
    1):USING "#":R :: L=5 :: F=0
910 DISPLAY AT(24,8)SIZE(1):USING "
    #":F :: DISPLAY AT(24,18)SIZE(2
    ):USING "#":L :: GOTO 340
920 SUB TITEL
930 CALL CLEAR
940 ! TEXAS
950 FOR X=91 TO 121
960 READ A$
970 CALL CHAR(X,A$)
980 NEXT X
990 DATA 0103030303030303,FC0405050
    406020C,0080404080000C12,03FF80
    C060381C0E,1921213D050504C2
1000 DATA BA8A8ABAA1A12142,0301,E13
```

Scan: S. Höltgen

```
     010180C070300,8C102040402OE0,00
     007E81817E0000,1F2040808087 8888
     ,F804020101E11111
1010 DATA 88878040201F,11E1010101F1
     1111,0000E09F8080403F,111131E10
     10204F8,0F1112222444488 8,000101
     0202040408
1020 DATA 9010202040408080,09111222
     244448F8,0102020409091224,F8080
     808C8484848,2448909F808080FF,48
     4848C80F01018F
1030 DATA 88888888888888F8,03040404
     04080809,C020202020101090,0A121
     21213202023,50484848C80404C4,26
     444444448888F8,6422222222211111F
1040 CALL HCHAR(3,5,91):: CALL HCHA
     R(3,6,92):: CALL HCHAR(3,7,93)
1050 CALL HCHAR(4,5,94):: CALL HCHA
     R(4,6,95):: CALL HCHAR(4,7,96)
1060 CALL HCHAR(5,5,97):: CALL HCHA
     R(5,6,98):: CALL HCHAR(5,7,99)
1070 CALL HCHAR(4,9,100)
1080 FOR X=1 TO 2
1090 CALL HCHAR(3,12+I,101):: CALL
     HCHAR(3,13+I,102)
1100 CALL HCHAR(4,12+I,103):: CALL
     HCHAR(4,13+I,104)
1110 CALL HCHAR(5,12+I,105):: CALL
     HCHAR(5,13+I,106)
1120 I=I+3 :: NEXT X
1130 CALL HCHAR(3,19,107):: CALL HC
     HAR(4,18,108):: CALL HCHAR(4,19
     ,109):: CALL HCHAR(5,18,110)
1140 CALL HCHAR(3,21,111):: CALL HC
     HAR(3,22,112):: CALL HCHAR(4,21
     ,113):: CALL HCHAR(4,22,114)::
     CALL HCHAR(5,22,115)
1150 CALL HCHAR(3,24,116):: CALL HC
     HAR(3,25,117):: CALL HCHAR(4,24
     ,118):: CALL HCHAR(4,25,119)::
     CALL HCHAR(5,24,120):: CALL HCH
     AR(5,25,121)
1160 CALL CHAR(128,"01572523211F070
     3031F3B130904080080EAA4C484F8E0
     C0E0F8DCC890201000")
1170 CALL CHAR(132,"01070503010FB7C
     3070F1F1B112020C080E0A0C080F0ED
     C3E0F0F8D888040403")
1180 CALL MAGNIFY(3)
1190 FOR S=1 TO 5
1200 CALL SPRITE(#S,128,13,89,73+R2
     )
1210 R2=R2+32
1220 NEXT S
1230 R2=0
1240 FOR S=6 TO 10
1250 CALL SPRITE(#S,128,13,137,73+R
     2)
1260 R2=R2+32
1270 NEXT S
1280 DISPLAY AT(15,11):"FROG-JOG" :
     : DISPLAY AT(24,9):"PRESS ANY K
     EY"
1290 RESTORE 1340
1300 FOR SOUND=1 TO 15 :: READ DA,S
     2,S3 :: CALL SOUND(DA,S2,2,S3,3
     )
1310 CALL KEY(0,K,S):: IF S=0 THEN
     1330
1320 SUBEXIT
1330 NEXT SOUND :: GOTO 1290
1340 DATA 200,698,880,200,880,1047,
     200,1047,1319,175,698,880,175,8
     80,1047,175,1047,1319
1350 DATA 210,1175,988,210,988,784,
     210,1047,880,220,523,659,220,88
     0,659,220,784,659
1360 DATA 230,523,659,240,440,523,4
     00,440,659
1370 SUBEND
1380 SUB SIEG
1390 RESTORE 1400 :: FOR SOUND=1 TO
     5 :: READ S2 :: CALL SOUND(200
     ,523,2,S2,3):: NEXT SOUND
1400 DATA 1047,880,784,880,1047
1410 CALL SOUND(300,523,2,659,3)::
     CALL SOUND(400,523,2,330,2):: S
     UBEND
```

Letters

Ähnlich den Spielen Scrabble und Lingua müssen aus bestimmten Buchstaben, unter Berücksichtigung strategischer Überlegungen, Wörter gebildet werden (Spectrum 48K)



„LETTERS" hat deutliche Züge eines Brettspiels, ließe sich jedoch ohne Computer nur schwer verwirklichen.

Für den Spielgebrauch ein paar Hinweise:

– Die Spieldauer beträgt im Durchschnitt ca. 1 bis 2 Std.

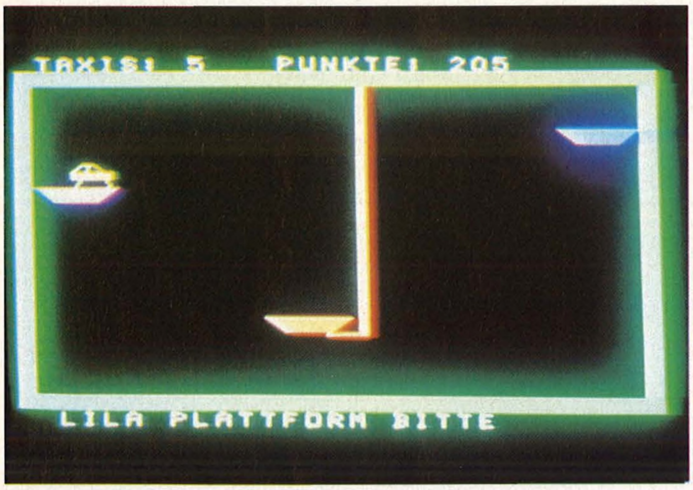
– Für ein augenschonendes Spielen ist an eine individuelle Spielbrett-

Scan: S. Höltgen


```
2870 aug=0
2880 RETURN
2890 IF MAX(xx(1),xx(2),xx(3),xx(
4),xx(5),xx(6))=5 GOTO 2930
2900 IF MAX(xx(1),xx(2),xx(3),xx(
4),xx(5),xx(6))<>3 THEN aug=0:GOT
O 2940
2910 FOR k=1 TO 6:IF xx(k)=3 THEN
 xx(k)=0 ELSE NEXT
2920 IF MAX(xx(1),xx(2),xx(3),xx(
4),xx(5),xx(6))<>2 THEN aug=0:GOT
O 2940
2930 aug=25
2940 RETURN
2950 PRINT #1,FNp$(lin(i),7+j*8);
USING "#######";lst(am,j,i)
2960 lst(am,j,18)=lst(am,j,18)+au
g
2970 GOSUB 2980:RETURN
2980 PRINT #1,FNp$(10,7+j*8);USIN
G "#######";lst(am,j,7)
2990 PRINT #1,FNp$(11,7+j*8);USIN
G "#######";lst(am,j,8)
3000 PRINT #1,FNp$(23,7+j*8);USIN
G "#######";lst(am,j,18)
3010 suman(am)=0:FOR i=1 TO anzsp
:suman(am)=suman(am)+lst(am,i,18)
:NEXT
3020 PRINT #1,FNp$(25,18);USING "
####";suman(am);
3030 RETURN
3040 REM *** accept-sound
3050 RESTORE 3070
3060 READ ton,tim:IF ton=0 AND ti
m=0 THEN RETURN ELSE SOUND 2,ton,
tim,7:                SOUND 2,ton*2,
12-tim,3:GOTO 3060
3070 DATA 190,10,239,10,142,5,142
,5,142,5,0,0
```

Space-Taxi

Als Führer eines Raketentaxis tragen Sie die Verantwortung, Ihre Fahrgäste sicher durch den Weltraum zu transportieren (TI-99/4A mit Extended BASIC und Joystick)



Ziel dieses Spieles ist es, Raumfahrer im Space-Taxi in möglichst kurzer Zeit von Plattform zu Plattform zu transportieren. Je schneller dies geschieht, desto mehr Punkte gibt es.

Das Taxi läßt sich mit dem Joystick 1 steuern. Die Steuerung erfolgt nach dem Raketenprinzip durch Impulse. Weiterhin wirkt eine Schwerkraft auf das Taxi, die auch ohne Steuerung zu einer Sinkgeschwindigkeit führt. Der Landevorgang wird oberhalb der jeweiligen Plattform durch Betätigen des Feuerknopfes eingeleitet. Mit ausgefahrenen Landekufen kann nicht mehr in horizontaler Richtung (links und rechts) gesteuert werden. Eine zu hohe Landegeschwindigkeit führt zum Bruch und damit zum Absturz des Fahrzeuges. Die Anweisungen über Abholung und Ziel des Fahrgastes stehen in der letzten Zeile.

Mit dem Erreichen eines Vielfachen von 500 (500, 1000, ...) beginnt eine neue Spielstufe. Wenn alle sechs Taxis den Tücken des Weltraums zum Opfer gefallen sind, ist das Spiel zu Ende. In der zweiten Spielstufe erscheint ein neues Bild. Wenn auch diese Stufe überwunden wurde (mehr als 1000 Punkte), beginnt wieder die erste Stufe, jedoch ist die Geschwindigkeit des Taxis doppelt so hoch. *Dieter Sundermann*

```
100 !
110 !
120 !
130 ! ***********************
140 !************************
150 !**                    **
160 !**    SPACE - TAXI    **
170 !**                    **
180 !**        VON         **
190 !**                    **
200 !**  JENS SUNDERMANN   **
210 !**                    **
220 !************************
230 ! ***********************
240 !
250 !
260 !
270 GOTO 300 :: CALL COINC :: CALL POSITION :: CALL COLOR :: CALL HCHAR :: CALL
CHAR :: CALL SPRITE :: CALL VCHAR
280 CALL DELSPRITE :: CALL KEY :: CALL MOTION :: CALL JOYST :: CALL SCREEN :: CA
LL CLEAR :: CALL LOCATE
```

```
290 CALL PATTERN :: CALL SOUND :: CALL MAGNIFY :: VE :: A$ :: B$ :: STUFE :: R :
: I :: ENDE :: SCORE1 :: LIVE :: !§P-
300 CALL CLEAR
310 CALL SCREEN(2)
320 CALL COLOR(5,16,2,6,16,2,7,16,2,8,16,2)
330 !
340 !***********************
350 !*                     *
360 !*    TITELBILD        *
370 !*                     *
380 !***********************
390 !
400 A$=" SS  PPP   AA   CC  EEE      S  S P  P A  A C  C E        S    P  P A  A C
 E        SS  PPP  AAAA C    EE  EEE    S P    A  A C    E"
410 A$=A$&"        S  S P    A  A C  C E        SS  P    A  A  CC  EEE"
420 B$="TTTTT  AA  X   X  I            T   A  A X   X  I          T   A   A  X X
 I         T   AAAA   X    I          T   A  A  X X   I "
430 B$=B$&"          T   A  A X   X  I          T   A  A X   X  I"
440 FOR I=1 TO 7
450 DISPLAY AT(4,1):SEG$(A$,197-I*28,I*28)
460 DISPLAY AT(19-I,1):SEG$(B$,1,I*28)
470 NEXT I
480 FOR I=1 TO 50 :: CALL SCREEN(RND*14+2):: NEXT I :: CALL CLEAR
490 CALL SCREEN(2):: CALL COLOR(14,4,4,12,16,15,1,16,2,2,16,2,3,16,2,4,16,2,5,16
,2,6,16,2,7,16,2,8,16,2)
500 !
510 !***********************
520 !*                     *
530 !*   BILDSCHIRMRAND    *
540 !*                     *
550 !***********************
560 !
570 CALL HCHAR(2,2,138,30)
580 CALL HCHAR(23,2,138,30)
590 CALL VCHAR(3,2,138,21)
600 CALL VCHAR(3,31,138,21)
610 !
620 !***********************
630 !*  DEFINITION DER     *
640 !*      CHARACTER      *
650 !***********************
660 !
670 A$=RPT$("F",16):: CALL CHAR(138,A$,140,A$)
680 CALL CHAR(136,"FF7F3F1F0F070301")
690 CALL CHAR(142,"FFFEFCF8F0E0C080")
700 CALL CHAR(96,RPT$("00",16)&RPT$("0F",16))
710 CALL CHAR(100,"FFFF"&RPT$("00",14)&"FFFF"&RPT$("00",14))
720 CALL CHAR(104,"00030303010070B0B03020202020000000808080000C0A0A0A080808080B0"
)
730 CALL CHAR(108,"0303030103050901021408000000000080808000804020008040E0")
740 CALL CHAR(112,"030303010101010101107"&RPT$("00",22))
750 CALL CHAR(92,"0800043505D40124004210950022240000000803058200A401280 48A20800B0
")
760 CALL CHAR(132,"0F122242FFFFFFFF3838100000000000080402010FFFFFFFF0E0E04")
770 CALL CHAR(128,"0F122242FFFFFFFF1010202020400000080402010FFFFFFFF040402020201"
)
780 CALL MAGNIFY(3)
790 !
800 ! *********************
810 !***********************
820 !**                   **
830 !**      1. STUFE     **
840 !**                   **
850 !***********************
860 ! *********************
870 !
880 VE=0.5 :: SCORE1=0 :: LIVE=5 :: STUFE=1
890 DISPLAY AT(1,1):"TAXIS: 5    PUNKTE: 0"
900 CALL DELSPRITE(ALL):: CALL SPRITE(#2,136,10,136,100)
910 CALL SPRITE(#4,136,14,69,17)
920 CALL SPRITE(#6,136,6.39,209)
930 CALL SPRITE(#3,140,10,136,116)
940 CALL SPRITE(#5,140,14,69,33)
950 CALL SPRITE(#7,140,6,39,225)
```

Scan: S. Höltgen


```
960 FOR I=8 TO 16
970 CALL SPRITE(#I,96+(I=16)*(-4),10,(I-7)*16,122)
980 NEXT I
990 CALL SPRITE(#1,132,12,70,70)
1000 CALL ROUT(ENDE,SCORE1,LIVE,3,500*STUFE,VE)
1010 STUFE=STUFE+1 :: IF ENDE=1 THEN 880
1020 !
1030 ! ******************
1040 !*********************
1050 !**                 **
1060 !**    2. STUFE     **
1070 !**                 **
1080 !*********************
1090 ! ******************
1100 !
1110 CALL DELSPRITE(ALL)
1120 R=8
1130 FOR I=1 TO 170 STEP 30
1140 R=R+2 :: CALL SPRITE(#R,96,7,I,125,10,0)
1150 CALL SPRITE(#R+1,96,7,171-I,130,-10,0)
1160 NEXT I
1170 CALL SPRITE(#1,132,12,70,70)
1180 CALL SPRITE(#2,136,10,100,150,#3,140,10,100,166)
1190 CALL SPRITE(#4,136,14,40,40,#5,140,14,40,56)
1200 CALL SPRITE(#6,136,6,140,16,#7,140,6,140,32,#8,100,15,125,35)
1210 CALL ROUT(ENDE,SCORE1,LIVE,3,500*STUFE,VE)
1220 IF ENDE=1 THEN 880
1230 STUFE=STUFE+1 :: VE=VE+.5 :: GOTO 900
1240 !§P+
1250 !
1260 ! ******************
1270 !*********************
1280 !**                 **
1290 !** HAUPTPROGRAMM   **
1300 !**                 **
1310 !*********************
1320 ! ******************
1330 !
1340 SUB ROUT(ENDE,SCORE1,LIVE,ANZ,COMP,VE)
1350 GOTO 1380 :: C :: SCORE :: T$(1):: T$(2):: T$(3):: XM :: YM :: PAT :: CNT :
: IND :: PAD1 :: PAD
1360 J :: Y1 :: X1 :: FA :: FX :: FY :: I :: Y :: X :: M :: N :: K :: L :: LY ::
 LX
1370 !§P-
1380 RANDOMIZE
1390 T$(1)="ORANGE"
1400 T$(2)="LILA"
1410 T$(3)="BLAUE"
1420 ENDE,XM,YM=0 :: PAT=132
1430 CNT=0 :: IND=1
1440 PAD1=PAD :: PAD=INT(RND*ANZ)+1 :: IF PAD=PAD1 THEN 1440
1450 IF IND=0 THEN DISPLAY AT(24,2):T$(PAD);" PLATTFORM BITTE" :: IND=1 :: CALL
DELSPRITE(#20):: GOTO 1490
1460 IF IND=1 THEN DISPLAY AT(24,2):"HEY TAXI,BITTE KOMMEN" :: IND=0 :: SCORE=30
0
1470 CALL POSITION(#PAD*2,Y,X)
1480 CALL MOTION(#20,0,0):: CALL SPRITE(#20,104,16,Y-14,X+23)
1490 CALL JOYST(1,M,N)
1500 XM=XM+M*VE*(ABS(PAT<>128))
1510 YM=YM-N*VE+.3
1520 CALL MOTION(#1,YM,XM)
1530 CALL COINC(ALL,C):: IF C THEN 2120
1540 CALL KEY(1,K,L):: IF L THEN 2080
1550 CALL POSITION(#1,Y,X)
1560 IF Y<17 OR Y>167 THEN 1610
1570 IF X<17 OR X>225 THEN 1610
1580 CALL COINC(ALL,C):: IF C THEN 2120
1590 SCORE=SCORE-1
1600 GOTO 1490
1610 !
1620 !
1630 !*********************
1640 !*                   *
1650 !*      CRASH        *
```

Scan: S. Höltgen


```
1660 !*                    *
1670 !*********************
1680 !
1690 !
1700 CALL PATTERN(#1,92)
1710 CALL SOUND(400,-5.76,0)
1720 CALL MOTION(#1,0,0)
1730 CALL POSITION(#1,Y,X)
1740 FOR I=Y TO 165 STEP 5
1750 CALL LOCATE(#1,I,X)
1760 CALL SOUND(-50,(800-I),0)
1770 NEXT I
1780 CALL DELSPRITE(#1)
1790 CALL SOUND(600,110,30,110,30,3000,30,-5,0)
1800 LIVE=LIVE-1
1810 IF LIVE<>-1 THEN 1900
1820 DISPLAY AT(24,1):" "
1830 DISPLAY AT(14,8):"ENDE"
1840 DISPLAY AT(19,4):"DRUECKE EINE TASTE"
1850 CALL KEY(0,M,N)
1860 IF N THEN 1870 ELSE 1850
1870 DISPLAY AT(14,1):" "
1880 DISPLAY AT(19,1):" "
1890 CALL DELSPRITE(ALL):: ENDE=1 :: SUBEXIT
1900 DISPLAY AT(1,7)SIZE(-3):LIVE
1910 FOR I=1 TO 200 :: NEXT I
1920 CALL SOUND(300,659,0)
1930 CALL SOUND(150,587,0)
1940 CALL SOUND(350,523,0)
1950 CALL SPRITE(#1,132,12,70,70)
1960 XM,YM=0 :: PAT=132
1970 GOTO 1490
1980 !
1990 !
2000 !*********************
2010 !*                   *
2020 !*    LANDEKLAPPEN   *
2030 !*     AUSFAHREN     *
2040 !*                   *
2050 !*********************
2060 !
2070 !
2080 PAT=130+(2+(PAT=132)*4)
2090 CALL SOUND(10,700,0)
2100 CALL PATTERN(#1,PAT)
2110 GOTO 1550
2120 !
2130 !
2140 !*********************
2150 !*                   *
2160 !*  UNTERPROGRAMM    *
2170 !*FUER SPRITEKOLLISION*
2180 !*                   *
2190 !*********************
2200 !
2210 !
2220 CALL MOTION(#1,0,0)
2230 IF PAT=132 THEN 1610
2240 CALL POSITION(#1,LY,LX)
2250 FOR I=2 TO ANZ*2 STEP 2
2260 CALL POSITION(#I,PY,PX)
2270 CALL COINC(#I,#1,16,K):: IF K AND(LY+10<PY AND LX>PX AND LX<PX+16)THEN 2300
2280 NEXT I
2290 GOTO 1610
2300 CALL SOUND(200,110,0)
2310 FA=0 :: IF IND=1 THEN 2330
2320 IF (I/2)=PAD THEN 2570
2330 IF (I/2)<>PAD THEN FA=1 :: GOTO 2480
2340 CALL POSITION(#I,Y,X)
2350 CALL POSITION(#1,Y1,X1)
2360 CALL SPRITE(#20,108,16,Y-11,X+8,0,-4)
2370 FOR I=1 TO (X1+6-X)/2
2380 CALL PATTERN(#20,112)
2390 FOR J=1 TO 10 :: NEXT J
```

```
2400 CALL PATTERN(#20,108)
2410 FOR J=1 TO 10 :: NEXT J
2420 NEXT I
2430 CALL DELSPRITE(#20)
2440 IF SCORE<0 THEN 2480
2450 SCORE1=SCORE1+SCORE
2460 DISPLAY AT(1,19):SCORE1
2470 IF SCORE1>COMP THEN SUBEXIT
2480 XM,YM=0 :: PAT=132
2490 CALL JOYST(1,M,N)
2500 IF N THEN 2510 ELSE 2490
2510 CALL SOUND(10,700,0)
2520 CALL PATTERN(#1,132)
2530 YM=YM-N/2+.3
2540 XM=XM+M/2*(ABS(PAT<>128))
2550 CALL MOTION(#1,YM,XM)
2560 IF FA THEN 1490 ELSE 1440
2570 CALL POSITION(#1,Y,X)
2580 CALL POSITION(#20,Y1,X1)
2590 CALL LOCATE(#20,Y1+3,X1)
2600 CALL MOTION(#20,0,-4)
2610 FOR I=1 TO (X1-X)/2
2620 CALL PATTERN(#20,108)
2630 FOR J=1 TO 10 :: NEXT J
2640 CALL PATTERN(#20,112)
2650 FOR J=1 TO 10 :: NEXT J
2660 NEXT I
2670 CALL DELSPRITE(#20)
2680 GOTO 2480
2690 !$P+
2700 SUBEND
```

Scan: S. Höltgen

Kurz und listig

Zehn kleine Listings: lauffähig auf allen Home-Computern und im Nu eingetippt

Fakultät

```
10 INPUT Z
20 LET F=1
30 FOR I=1 TO Z
40 LET F=F*I
50 NEXT I
60 PRINT F
```

Mit dem Namen Fakultät haben Mathematiker eine Rechenvorschrift bezeichnet, die quasi aus einer Mücke einen Elefanten macht. Mit anderen Worten: Sie läßt relativ kleine Zahlen ins Gigantische wachsen. Dazu müssen nur, beginnend mit der 1, nacheinander alle ganzen Zahlen bis zu der ausgewählten miteinander multipliziert werden. n! ist übrigens die abkürzende Schreibweise für die Fakultät einer ganzen Zahl n.

Nach RUN ist eine positive ganze Zahl einzugeben, von der dann die Fakultät berechnet wird.

Primzahl – ja oder nein?

```
10 INPUT Z
20 FOR I=2 TO SQR(Z)
30 IF Z/I=INT(Z/I) THEN GOTO 70
40 NEXT I
50 PRINT "PRIMZAHL"
60 GOTO 10
70 PRINT "KEINE PRIMZAHL"
80 GOTO 10
```

Noch längst nicht jedes Geheimnis ist um sie gelüftet und sie geben immer wieder neue Rätsel auf: die Primzahlen. Zur Erinnerung: eine Primzahl läßt sich nur durch 1 und sich selbst ohne Rest teilen. Die größte derzeit bekannte ist $2^{132049} - 1$. Aber versuchen Sie bitte nicht mit dem obigen Programm eine größere zu finden. Dazu muß nämlich schon ein Großrechner her, und auch der ist dann lange Zeit für nichts anderes zu sprechen.

Das Programm erwartet nach RUN die Eingabe der zu testenden Zahl und gibt am Schluß das Ergebnis der Untersuchung auf dem Bildschirm aus.

Beim TI-99/4A muß GOTO in Zeile 30 weggelassen werden.

Toto

```
10 FOR I=1 TO 10
20 FOR J=1 TO 11
30 PRINT INT(RND(1)*3)
40 NEXT J
50 PRINT
60 NEXT I
```

Die größten Quoten ergeben sich beim Fußballtoto ja bekanntlich, wenn die Spiele tatsächlich so ausgehen, wie sie von einem Fußball-Laien getippt wurden. Da ein Home-Computer von Hause aus nicht gerade ein Experte in Sachen „rundes Leder" ist, könnte man ihn ja mal befragen.

Scan: S. Höltgen

Das obige Programm wirft gleich alle zehn Tipreihen der 11er-Wette für einen ganzen Tipschein aus.

Bei einigen Home-Computern muß RND(1) durch RND ersetzt werden.

Roulette

```
10 PRINT INT(RND(1) * 37)
20 GOTO 10
```

Solange an einem Roulettetisch keine Manipulationen vorgenommen wurden, rollt die Kugel mit der gleichen Wahrscheinlichkeit auf jede der Zahlen zwischen 0 und 36.

Dieses Programm liefert ununterbrochen Roulettezahlen. Erkennen Sie eine Systematik? Vielleicht sind ja doch die Permanenzen der nächsten Spielbank darin versteckt.

Bei einigen Home-Computern muß RND(1) durch RND ersetzt werden.

Von Woche zu Woche

```
10 DIM J(12)
20 FOR I=1 TO 12
30 READ J(I)
40 NEXT I
50 DATA 31,28,31,30,31,30
60 DATA 31,31,30,31,30,31
70 LET L=1
80 INPUT T
90 INPUT M
100 LET S=T+L
110 IF M=0 THEN GOTO 150
120 FOR I=1 TO M-1
130 LET S=S+J(I)
140 NEXT I
150 LET W=INT(S/7)+1
160 LET R=S-INT(S/7) * 7
170 PRINT W,R
```

Die Durchnumerierung aller Wochen eines Jahres bringt mehr Ordnung in einen Terminkalender. Ist gerade keiner zur Hand, kann dieses Programm weiterhelfen: es gibt zu einem beliebigen Datum die Wochenzahl an.

Einzugeben ist nach RUN das Datum in der Reihenfolge Tag dann Monat, also zum Beispiel 12 (RETURN) dann 7 (RETURN) für den 12. Juli. In der Zeile 70 befindet sich verschlüsselt der Wochentag des letzten Tages des Vorjahres, also für 1985 der 31. Dezember 1984. 0 steht dort für Sonntag (eine neue Woche beginnt für das Programm immer am Sonntag), 1 für Montag und so weiter bis zur 6 für Sonnabend. Für andere Jahre ist Zeile 70 entsprechend zu ändern, bei Schaltjahren außerdem die zweite Zahl in Zeile 50, sie gibt die Zahl der Tage im Februar an. Neben der Wochenzahl wird vom Programm noch der Wochentag am eingegebenen Datum mitgeteilt, als Zahl zwischen 0 und 6 in der oben erwähnten Verschlüsselung.

Beim TI-99/4A muß GOTO in Zeile 110 weggelassen werden. Der ZX81 kennt READ und DATA nicht, dort muß man J(1) bis J(12) mit LET-Anweisungen die 12 Werte zuweisen.

Fahrenheit

```
10 PRINT "FAHRENHEIT",
"CELSIUS"
20 INPUT F
30 LET C=(F-32) * 5/9
40 LET C=INT(C+0.5)
50 PRINT F,C
60 GOTO 20
```

Zur Messung von Temperaturen gibt es ja neben der Celsius-Skala unter anderem auch die Fahrenheit-Skala, die zum Beispiel in den Vereinigten Staaten Verwendung findet. Bevor man über den großen Teich in die Neue Welt fliegt, sollte man schon wissen, ob bei 32° eine Badehose das passende Kleidungsstück ist oder einem bereits die Ohren abfrieren.

Das Programm rechnet nach RUN eingegebene Fahrenheit-Werte in die Celsius-Skala um und rundet danach auf ganze Zahlen.

Einmaleins

```
10 FOR I=1 TO 30
20 FOR J=1 TO 10
30 PRINT I * J;" ";
40 NEXT J
50 PRINT
60 NEXT I
```

Die Zeiten, in denen Grundschüler mit Durchschnittsmengen vertrauter waren als mit dem kleinen Einmaleins, dürften wohl vorbei sein. Trotzdem beträgt für die meisten menschlichen Gehirne die Zeit vom Input von Aufgaben wie 7 * 28 bis zum Output des Ergebnisses vermutlich einige Sekunden.

Als Stütze liefert das Programm die Vielfachen bis zum Zehnfachen der Zahlen 1 bis 30.

Zufallsgenerator

```
10 LET ZZ=0
20 LET ZZ=75 * (ZZ+1)-1
30 LET ZZ=ZZ-INT(ZZ/655537)
* 65537
40 PRINT ZZ/65536
50 GOTO 20
```

Nichts bleibt dem Zufall überlassen, auch der Zufall sich selbst nicht. Damit Home-Computer Zufallszahlen liefern können, die scheinbar ungeordnet verteilt sind, gibt es viele Verfahren. Eines der kürzesten wird in diesem Programm demonstriert. Es beruht darauf, daß sich durch eine einfache Rechnung für ZZ nacheinander alle Zahlen zwischen 0 und 65535 ergeben, und zwar in kaum vorhersehbarer Reihenfolge, es sei denn, man kennt die Rechenvorschrift.

Das Programm liefert nach RUN eine Folge von Zufallszahlen zwischen 0 (einschließlich) und 1 (die 1 selbst nicht). Durch Ändern der Zeile 10 kann man die Folge an einer anderen Stelle starten.

Nicht nachtragend

```
10 PRINT "WUERDEN SIE DIESE
BELEIDIGUNG EVENTUELL
ZURÜCKNEHMEN ?"
20 INPUT A$
30 IF A$="NEIN" THEN PRINT
"DANN IST DIE SACHE FUER
MICH ERLEDIGT"
40 IF A$="JA" THEN PRINT "ZU
GUETIG"
```

Wenn Sie Ihren Home-Computer mal richtig verflucht haben, wenn er Ihnen allzu viele Fehlermeldungen vorgesetzt hat, dann nimmt er Ihnen das nicht übel, wie dieses Programm beweist.

Bei Atari-Computern muß noch die Programmzeile 5 DIM A$(4) eingefügt werden.

Vollkommen

```
10 LET Z=1
20 LET S=0
30 LET Z=Z+1
40 FOR I=1 TO Z-1
50 IF INT(Z/I)=Z/I THEN LET
S=S+I
60 NEXT I
70 IF Z=S THEN PRINT Z;" VOLL-
KOMMENE ZAHL"
80 GOTO 20
```

Eine Zahl bezeichnet man als vollkommen, wenn sie gleich der Summe ihrer Teiler ist (zum Beispiel 6 = 1 + 2 + 3). Die Zahl selbst darf dabei nicht mitaddiert werden. Das Programm spürt solche Exoten im Zahlendschungel auf. *br*

Scan: S. Höltgen

sen Sie diese Grafik. Der Pfeil unterhalb der 1 ist für genauere Auskünfte bestimmt.
Auch das ist kinderleicht: Sie fahren mit den Cursortasten (links oder rechts) unter das gewünschte Datum und drücken RETURN. Jetzt erhalten Sie nähere Auskünfte zu dem Tag, entweder auf Bildschirm oder Plotter.
Bei dem Anfangspunkt Zeitspanne gibt es noch sehr viel mehr Darstellungen, zum Beispiel:
Ausgaben bei einem Händler
Ausgaben in einer Zielgruppe (Getränke)
Gesamtausgaben (von ... bis ...) usw. *Günter Liesner*

Listing auf Seite 75.

Variablenlister

Ein Hilfsprogramm, das bei der Fehlersuche hilft, indem es sämtliche bisher benutzten Variablen und ihre zugehörigen Werte auf dem Bildschirm oder Drucker auflistet (16K und 48K)

Wer kennt das nicht. Man hat ein langes Programm geschrieben und es funktioniert nicht richtig. Oft liegt der Fehler daran, daß man irgendwo eine Variable falsch belegt oder verwechselt hat. Hier springt nun mein Programm ein, denn es bewirkt, daß alle bisher benutzten Variablen mitsamt ihren jetztigen Werten entweder auf den Bildschirm oder auf den Drucker ausgegeben werden, so daß man den Fehler leichter finden kann. Da das Programm in Maschinensprache geschrieben ist, werden auch die Zeilen des eigenen Programmes nicht verändert, wenn man mein Programm dazulädt.

Eingabe und Benutzung

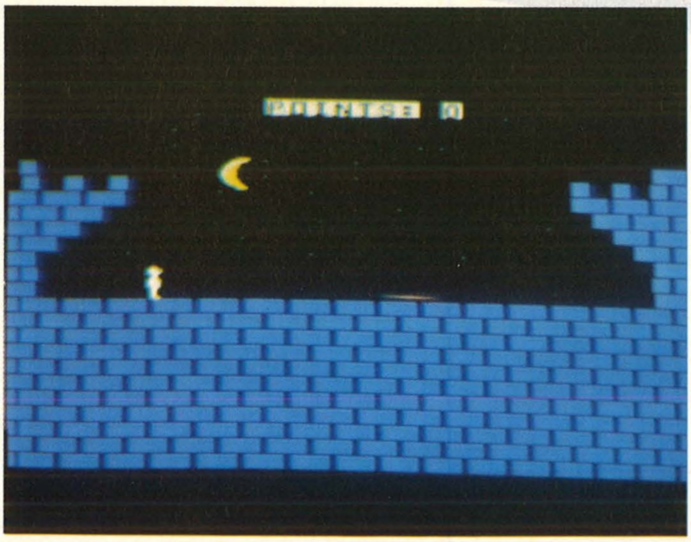
Von dem Programm gibt es zwei Versionen. Eine für den Spectrum mit 16K, eine für den mit 48K. Man tippt nun das BASIC-Listing für seine Version ab und startet das Programm mit RUN. Darauf hin muß man seinen Kassettenrecorder aufnahmebereit machen, da nun das Maschinenspracheprogramm gesaved wird. Dann muß man zurückspulen, und der Computer überprüft die Aufnahme mit VERIFY.

Benutzung des Programmes:
Mein Programm kann jederzeit zu einem BASIC-Programm dazugeladen werden.
Dazu muß man folgendes gun:
a) 16K
Geben Sie bitte folgendes ein:
CLEAR 31699:LOAD"" CODE
b) 48K
Geben Sie bitte folgendes ein:
CLEAR 64499:LOAD"" CODE
Nun laden Sie bitte das Maschinencodeprogramm, das Sie auf Kassette abgespeichert haben, achten Sie darauf, daß Sie die richtige Version für Ihren Computer haben. Bevor Sie den Variablenlister benutzen, müssen Sie zuerst Ihr eigenes Programm starten. Der Aufruf des Variablenlisters geschieht folgendermaßen:
a) 16K
Ausgabe auf Bildschirm:
RANDOMIZE USR 31700
Ausgabe auf Drucker:
RANDOMIZE USR 31704
b) 48K
Ausgabe auf Bildschirm:
RANDOMIZE USR 64500
Ausgabe auf Drucker:
RANDOMIZE USR 64504

Martin Pfundheller

Listing auf Seite 78.

Hunch-Back



Wunderschöne Eisblumen sind in einer alten Burg versteckt. Fallgruben, Feuerbälle und gefährliche Pfeile erschweren den Weg zum Ziel (TI-99/4A Extended-BASIC und Joystick)

Das folgende Programm ist eine mehr oder weniger stark abgewandelte Version des populären Commodore-Spiels, das an die Grafik seines Vorbilds nicht ganz, aber immerhin recht gut herankommt. Es beginnt alles ganz harmlos. Man sieht auf dem Bildschirm eine Burgmauer, rechts und links einen Turm, ein weißes Männchen und einen Pfeil, der sich von rechts nach

Scan: S. Höltgen

links bewegt. Ihre Aufgabe ist es nun, ihre Spielfigur unbeschadet auf die andere Seite des Bildschirms zu bringen. Bei diesem Unterfangen können Sie sich natürlich nach links oder rechts bewegen, hüpfen oder sich auf den Boden legen.
Hierbei sind die folgenden Funktionen zu beachten:
Joystick links oder rechts = Männchen links oder rechts
Joystick nach unten = Hinlegen
Feuerknopf = Hüpfen.
Beim Hüpfen ist außerdem darauf zu achten, daß Sie während des Feuerknopf-Drückens in Bewegung sind. Stehen Sie und drükken den Button, so geschieht überhaupt nichts. Natürlich gilt auch hier: Übung macht den Meister.

Haben Sie nun das andere Bildschirmende erreicht, geht's ins Burgmauer-Bild, das natürlich schwerer ist. So steigt zum Beispiel die Anzahl der Pfeile und es kommen Löcher hinzu und so weiter. Ganze achtmal müssen Sie diese Burgmauer-Qualen über sich ergehen lassen, bevor Sie mit einem Freimännchen belohnt werden und in den nächsten Sektor kommen.

Ohne Fleiß kein Preis

Hier gilt es den Turm der Burg zu erklimmen. Dies wäre weiter nicht schlimm, würden nicht von oben auf einmal Feuerbälle herunterfallen, mit denen eine Berührung nicht anzuraten ist. Natürlich meistert ein Super-Video-Spieler wie Sie auch diese Hürde und gelangt somit ins letzte Bild.

Man befindet sich nun in der Burg selbst, besser gesagt in einem Raum mit Eisblumen, die nur darauf warten gepflückt zu werden. Sie stehen vorerst in der Mitte des Raumes und machen gar nichts, aber nur solange Sie ihren Joystick ruhighalten. Bewegen Sie ihn nämlich in irgendeine Richtung, beginnt das Männchen zu laufen. Zwar ist es ihnen per Joystick anheimgestellt, seine Richtung zu ändern

EPSON zum Thema Drucker.

Mit der neuen LX-Serie ohne am falschen Ende

Scan: S. Höltgen

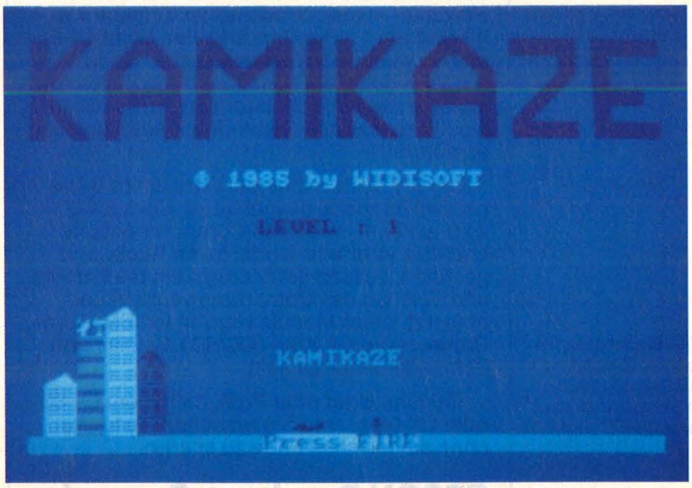
und ihn somit nicht gegen eine Mauer laufen zu lassen, aber stoppen läßt er sich trotzdem nicht. Pro Eisblume werden Ihnen 100 Punkte gutgeschrieben und sind alle eingesammelt, so geht das Ganze wieder von vorne los.

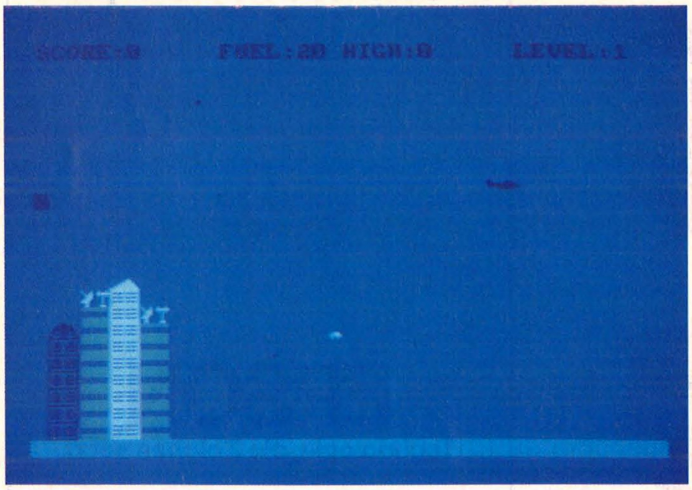
Am Anfang des Spieles können Sie durch die Taste „A" eine Spielanleitung abrufen. Durch Drücken des Firebutton wird das Spiel gestartet. Mein persönlicher Highscore liegt bei 156 400 Punkten. Wer ihn eines Tages mal brechen sollte, soll mir doch bitte schreiben. Aber jetzt viel Spaß beim Spielen. *Michael Schneller*

Das dazugehörende Listing befindet sich auf Seite 79.

Kamikaze

Schießen Sie sich eine Landebahn auf dem Spielfeld des Schneider CPC frei, ohne dabei mit den Häusern zu kollidieren





In dem Spiel Kamikaze sind Sie Pilot eines Düsenflugzeuges und müssen versuchen, auf dem Erdboden zu landen. Dazu ist es aber nötig, die Häuser, die im Weg stehen, niederzureißen, um eine Landebahn zu haben und nicht an die Häuserblocks zu stoßen. Zu diesem Zweck haben Sie den Fire-Knopf des Joysticks, mit dem Sie eine Bombe abwerfen können, die einen Teil des Hauses zerstört. Erst, wenn alle Häuser niedergerissen sind, oder Sie schon vorher durch geschicktes Manövrieren des Flugzeugs zwischen den Häuserblocks gelandet sind, haben Sie die Aufgabe bestanden.

Damit es nicht allzu schwer ist, ist es möglich, bei einer bevorstehenden Kollission mit einem Haus das Flugzeug nach oben zu steuern. Die Anzahl der möglichen Aufstiege wird als FUEL am Bildschirm angezeigt.

Hinweis: Alle Bewegungen können mit dem Joystick oder mit der Tastatur gemacht werden. Dabei ersetzt Space-Taste den FIRE-Knopf und die Cursortasten die Joystickbewegung.

Nach dem Starten des Spiels erscheint das Titelbild. Jetzt können Sie mit dem Joystick (bzw. mit den Cursortasten) einen Level von eins bis drei einstellen oder sofort mit dem FIRE-Knopf der Space-Taste das Spiel starten. Während des Spieles ist es möglich, das Flugzeug nach oben oder nach unten zu steuern. Nach der Landung starten Sie unter schwierigen Bedingungen erneut, falls Sie es nicht geschafft haben, müssen Sie von vorne anfangen. *Matthias Withopf*

Das dazugehörende Listing finden Sie auf Seite 82.

```
 150 DATA 3669,"F5CF14536368 6C65
6966653A20537472696E6766656C643A
204E756D2E2D46656C643A2044494D3A
20"
 200 REM Saven
 210 CLS : PRINT "Ich SAVE nun d
en Maschinencode  ab, mit dem Si
e spaeter arbeiten"'"Bitte mache
n Sie Ihren Recorder aufnahmeber
eit!"
 220 SAVE "Var.-list"CODE 31700,
365
 230 PRINT "Ich ueberpruefe nun,
 ob die        Aufnahme gelungen is
t, bitte       spulen Sie zum Begin
 zurueck,      und lassen das Band
abspielen.     Sollte 'TAPE LOADING
 ERROR' er-    scheinen, so geben S
ie bitte       GOTO 200 ein, damit
eine neue      Aufnahme erfolgt."
 240 VERIFY ""CODE
 250 PRINT "Geschaft, viel Spass
 noch!!!!!": STOP
```

16K-Version

```
   4 CLEAR 64499
   5 LET f=64500
   6 RESTORE
  10 FOR i=1000 TO 1050 STEP 10
  11 READ t: READ a$
  12 LET s=0: PRINT i,
  15 FOR x=1 TO LEN a$ STEP 2: L
ET a=(CODE a$(x))-48: LET b=(COD
E a$(x+1))-48
  20 IF a>16 THEN LET a=a-7
  30 IF b>16 THEN LET b=b-7
  40 LET p=a*16+b: POKE f,p: LET
 f=f+1: LET s=s+p
  50 NEXT x
  60 IF s<>t THEN BEEP 1,1: PRIN
T FLASH 1;"Fehler": STOP
  90 PRINT "OK": NEXT i
 200 REM Saven
 210 CLS : PRINT "Ich SAVE nun d
en Maschinencode  ab, mit dem Si
e spaeter arbeiten"'"Bitte mache
n Sie Ihren Recorder aufnahmeber
eit!"
 220 SAVE "Var.-list"CODE 64500,
365
 230 PRINT "Ich ueberpruefe nun,
 ob die        Aufnahme gelungen is
t, bitte       spulen Sie zum Begin
 zurueck,      und lassen das Band
abspielen.     Sollte 'TAPE LOADING
 ERROR' er-    scheinen, so geben S
ie bitte       GOTO 200 ein, damit
eine neue      Aufnahme erfolgt."
 240 VERIFY ""CODE
 250 PRINT "Geschaft, viel Spass
 noch!!!!!": STOP
1000 DATA 6889,"3E0218023E03CD01
162A4B5C3E0DD73E0DD77EFE80C8FEE1
382FD680E5F50166FD02033E0D021137
FD010A00CD3C20F1D73E06D71166FDED
535D5CCDFB24CDE3"
1010 DATA 8280,"2D011300E1ED4A18
C3FEC13817E5F51141FD010C00CD3C20
F1D660D73E24D73E0DD71858FEA1383E
E5D640F5D70166FDF102E123E57EFEE1
3827D680F5D7F103"
1020 DATA 7217,"02033E0D021166FD
ED535D5C3E3AD73E06D7CDFB24CDE32D
E1010600ED4AC300FCF5D70318CAFE81
3848E5F5114DFD010B00CD3C20F1D620
D73E0DD71158FD01"
1030 DATA 7774,"0500CD3C20E1235E
23562346EBED5A133E28E5D5D778E14E
234623E5F5CD1B1A3E2CD7F14710EE3E
08D73E29D7E1E1C300FCFE802001C9FE
613826E5F5D73E3A"
1040 DATA 6718,"D73E06D7F10166FD
02033E0D020BED435D5CCDFB24CDE32D
010600E1ED4AC300FC444DD73E24D73E
3AD73E06D7235E2356234 4DED5A0AD7
1B037AB3CA00FC18"
1050 DATA 3669,"F5CF14536368 6C65
6966653A20537472696E6766656C643A
204E756D2E2D46656C643A2044494D3A
20"
```

48K-Version

TI: Hunch-Back

```
10 ! *     TI-HUNCH BACK      *
20 ! *                        *
30 ! *     TI99+EXT.BASIC     *
35 ! *                        *
40 ! *       (C) 1984         *
45 ! *                        *
50 ! * MICHAEL  SCHNELLER     *
55 ! *                        *
60 ! *    KARWENDELWEG 6      *
62 ! *                        *
65 ! *     8901 KISSING       *
70 ! *                        *
75 ! *   TEL.: 08233/5412     *
80 ! *                        *
85 ! ************************
100 CALL CHAR(33,"FFFFFFFFFFFFFFFF",34,"
3C3C3C3C3C3C3C3C",35,"0103070F1F3F7FFF",
36,"FFFEFCF8F0E0C080")
101 CALL CHAR(37,"80C0E0F0F8FCFEFF",38,"
FF7F3F1F0F070301",39,"0000FFFFFFFF")
105 CALL CHAR(96,"7F7F7F7F7F7F7F00FEFEFE
FEFEFEFE00")
110 CALL CHAR(40,"1818407E1A181838"):: C
ALL COLOR(2,6,2,9,5,2):: M=5 :: CALL SCR
EEN(2):: CALL CLEAR
115 CALL CHAR(100,"0103030100010303030 30
10101010101C0A0E0C080C0E0E0E0C08080808
0E0")
116 P,DU=0 :: CALL MAGNIFY(3):: FOR I=3
TO 8 :: CALL COLOR(I,2,15):: NEXT I :: R
```

Scan: S. Höltgen


```
ANDOMIZE :: CALL CHAR(140,RPT$("7F",16)&
RPT$("FC",16))
117 CALL CHAR(124,"000000000000000000000
048243F24480000000000000000000000406FF0
604")! PF. RECHTS
118 CALL CHAR(128,"00000000105070F7D0F0
73D0F07030000000000080E0BCF0E0FCB8E0E0FCC
")! BOLLEN
120 CALL CHAR(104,"0103030100010303030 30
10101020406C0A0E0C080C0E0F0ECE0C0C020201
018")
130 CALL CHAR(112,RPT$("0",26)&"07FE85"&
RPT$("0",26)&"EE3EE6")! LIEGEND
135 CALL CHAR(116,RPT$("0",22)&"2060FF60
20"&RPT$("0",22)&"0912FC1209",98,"8")! P
FEIL
136 CALL COLOR(1,4,1):: Q=37
140 CALL CHAR(120,"030F1F3E7E7CFCFCFCFC7
C7E3E1F0703E0C00000000000000000000000008
0C")
142 GOSUB 20000
145 CALL VCHAR(7,1,96,18):: CALL VCHAR(6
,2,96,19):: FOR I=5 TO 23 STEP 2 :: CALL
 HCHAR(I,31,96):: CALL HCHAR(I+1,31,97):
: CALL HCHAR(I,32,97)
146 CALL HCHAR(I+1,32,96):: CALL HCHAR(I
,2,97):: CALL HCHAR(I+1,2,96):: NEXT I
150 DISPLAY AT(6,1):" ` `
     ` ` `a`a                     `a`aa`a
                         `a``a
            `a"
155 DISPLAY AT(10,1):"a
         `"
160 A$="a`a`a`a`a`a`a`a`a`a`a`a`a`a`" ::
 B$="`a`a`a`a`a`a`a`a`a`a`a`a`a`a"
165 FOR I=14 TO 23 STEP 2 :: DISPLAY AT(
I,1):A$ :: DISPLAY AT(I+1,1):B$ :: NEXT
I :: DISPLAY AT(24,1):A$
170 CALL HCHAR(1,3,40,M-1):: DISPLAY AT(
1,11):"POINTS:";P
175 FOR I=1 TO 20 :: CALL HCHAR(INT(10*R
ND)+2,INT(20*RND)+7,98):: NEXT I :: CALL
 SPRITE(#12,120,11,32,81):: IF DU=1 THEN
 185
178 DISPLAY AT(24,1):A$ :: BEG=11
179 FOR I=1 TO 5 :: FOR S=BEG TO BEG+9 :
: CALL SOUND(-5,S^2,0):: NEXT S :: BEG=B
EG+5 :: NEXT I :: FOR I=1 TO 100 :: NEXT
 I
180 CALL SOUND(500,262,0):: CALL SOUND(3
00,262,0,330,0):: FOR I=1 TO 30 :: CALL
SOUND(-10,262,I,330,I,392,I):: NEXT I
185 CALL SPRITE(#1,100,16,89,Q,#2,116,7,
89,220):: CALL MOTION(#2,0,-15)
190 GOSUB 31999 :: P=P+100*M :: DISPLAY
AT(1,19):P
200 CALL SPRITE(#1,100,16,89,Q,#2,116,7,
89,249,#3,124,7,80,120):: CALL MOTION(#2
,0,-10,#3,0,17)
210 GOSUB 31999 :: P=P+200*M :: DISPLAY
AT(1,19):P
220 CALL SPRITE(#1,100,16,89,Q,#2,116,13
,89,120,#3,116,13,89,240,#4,124,13,77,12
0)
225 CALL MOTION(#2,0,-15,#3,0,-15,#4,0,2
1):: GOSUB 31999 :: P=P+500*M :: DISPLAY
 AT(1,19):P
230 CALL SPRITE(#1,100,16,89,Q,#4,116,7,
89,120,#5,116,7,89,240,#2,124,13,80,50,#
3,124,13,80,180)
235 CALL MOTION(#4,0,-12,#5,0,-12,#2,0,1
9,#3,0,19):: GOSUB 31999
240 P=P+1000*M :: DISPLAY AT(1,19):P
245 CALL HCHAR(14,12,32,2):: CALL HCHAR(
15,12,32,2):: CALL HCHAR(14,18,32,2):: C
ALL HCHAR(15,18,32,2):: CALL HCHAR(14,24
,32,2)
250 CALL HCHAR(15,24,32,2):: CALL SPRITE
(#2,140,2,102,89,#3,140,2,102,137,#4,140
,2,102,185,#1,100,16,89,Q)
260 GOSUB 31999 :: P=P+1200*M :: DISPLAY
 AT(1,19):P
265 CALL SPRITE(#1,100,16,89,Q,#2,140,2,
102,89,#3,140,2,102,137,#4,140,2,102,185
)
270 CALL SPRITE(#5,116,13,85,249,0,-25):
: GOSUB 31999 :: P=P+2000*M :: DISPLAY A
T(1,1):P
275 CALL SPRITE(#1,100,16,89,Q,#2,140,2,
102,89,#3,140,2,102,137,#4,140,2,102,185
)
280 CALL SPRITE(#5,116,13,85,249,0,-20,#
6,124,13,77,120,0,15):: GOSUB 31999
285 P=P+2000*M :: DISPLAY AT(1,19):P
290 CALL SPRITE(#1,100,16,89,Q,#2,140,2,
102,89,#3,140,2,102,137,#4,140,2,102,185
)
295 CALL SPRITE(#5,116,13,85,249,#6,116,
13,85,130,#7,124,13,77,120)
300 CALL MOTION(#5,0,-20,#6,0,-20,#7,0,2
0):: GOSUB 31999 :: P=P+3000*M :: DISPLA
Y AT(1,19):P :: CALL DELSPRITE(ALL)
305 CALL HCHAR(5,1,32,640):: M=M+1 :: CA
LL HCHAR(1,3,40,M-1):: A$="`a`a`a`a`a`a`
a`a`a" :: B$="a`a`a`a`a`a`a`a`a`"
310 FOR I=1 TO 20 :: CALL SOUND(1,110,0)
:: CALL SOUND(1,220,0):: NEXT I :: FOR I
=0 TO 30 STEP 4 :: CALL SOUND(-100,160,I
):: NEXT I
315 DISPLAY AT(5,1):"      `a  `a  `a  `a
  `a          a`  a`  a`  a`  a`"
320 DISPLAY AT(7,6):A$ :: DISPLAY AT(8,6
):B$ :: DISPLAY AT(9,8)SIZE(14):A$ :: DI
SPLAY AT(10,8)SIZE(14):B$
325 FOR I=11 TO 23 STEP 2 :: DISPLAY AT(
I,10)SIZE(10):A$ :: DISPLAY AT(I+1,10)SI
ZE(10):B$ :: NEXT I
330 FOR I=1 TO 7 :: CALL HCHAR(INT(23*RN
D)+2,INT(5*RND)+1,98):: NEXT I :: FOR I=
1 TO 10 :: CALL HCHAR(INT(23*RND)+2,INT(
9*RND)+24,98):: NEXT I
340 CALL SPRITE(#15,120,11,17,225)
345 FOR I=0 TO 30 :: CALL SOUND(-100,110
,I):: CALL SOUND(-100,220,I):: NEXT I
350 X=16 :: Y=23 :: FOR I=2 TO 6 :: CALL
 SPRITE(#I,128,2,200,57+I*16):: NEXT I
355 FOR I=2 TO 6 :: CALL MOTION(#I,(30*R
ND)+30,0):: NEXT I :: CALL SPRITE(#1,100
,16,Y*8-7,X*8-7)
360 CALL JOYST(1,X1,Y1):: X=X+X1/2 :: Y=
Y-Y1/4 :: IF X>20 THEN X=20 ELSE IF X<12
 THEN X=12
365 IF Y>23 THEN Y=23 ELSE IF Y<10 THEN
500
370 CALL LOCATE(#1,Y*8-7,X*8-7):: CALL C
OINC(ALL,E):: IF E=-1 THEN 400 ELSE 360
400 CALL DELSPRITE(#2,#3,#4,#5,#6):: CAL
L PATTERN(#1,112):: FOR I=0 TO 30 STEP 2
 :: CALL SOUND(-100,110,I,220,I):: NEXT
I
410 CALL MOTION(#1,127,0):: FOR I=1 TO 1
0 :: NEXT I :: CALL DELSPRITE(#1):: M=M-
1 :: CALL HCHAR(1,3,32,5):: IF M=0 THEN
32600
420 CALL HCHAR(1,3,40,M-1):: GOTO 350
500 CALL DELSPRITE(ALL):: FOR I=0 TO 30
STEP 1.5 :: CALL SOUND(-100,-5.56,I):: N
EXT I :: P=P+1000*M :: DISPLAY AT(1,19):
P
510 CALL HCHAR(2,1,32,736):: CALL CHAR(4
1,"",42,"",43,""):: CALL CHAR(44,"182442
8181422418"):: CALL COLOR(2,14,2,9,13,2)
515 DISPLAY AT(2,1):"`a`a`a`a`a`a`a`a`a`
a`a`a`a`aa`          a`          a``a
       `a           `aa`          a`
        a`"
520 DISPLAY AT(6,1):"`a   a`
      a`a`a`a`a          `a`a`a`a`a`a`a
  a`        a`        a`a`   `a
 `a       `a"
525 DISPLAY AT(10,1):"`a   a`a`a`a`   a`
       a`a`          `a          `a`a
            a`       a`a`
  `a       `a"
530 DISPLAY AT(14,1):"`a`a`        `a
       a`a`         a`            `a`a
        `a          a`a`          a`
          `a"
```

```
535 DISPLAY AT(18,1):"`a                `a
         a`a`      `a`a`a`            `a`a
           `a`a`a`a`a      a`a`        a`
              `a"
540 DISPLAY AT(22,1):"`a                `a
         a`a`            a`           `a`a
`a`a`a`a`a`a`a`a`a`a`a`a`a"
543 EG,ST,TOT=0
545 IF TOT=1 THEN 600
550 FOR I=1 TO 20 :: CALL HCHAR(INT(21*R
ND)+3,INT(24*RND)+5,97):: NEXT I
555 CALL CHAR(61,"995A3CFFFF3C5A99"):: F
OR I=1 TO 20
560 XE=INT(24*RND)+5 :: YE=INT(21*RND)+3
 :: CALL GCHAR(YE,XE,H):: IF H>95 THEN 5
60 ELSE IF H=61 THEN 560
565 CALL HCHAR(YE,XE,61):: NEXT I
600 X=16 :: Y=12 :: CALL GCHAR(Y,X,H)::
IF H=97 THEN CALL HCHAR(Y,X,32)ELSE CALL
 SPRITE(#1,40,16,Y*8-7,X*8-7):: XR,YR=0
610 CALL JOYST(1,X1,Y1):: IF X1+Y1=0 THE
N 650 ELSE XR=X1/4 :: YR=-Y1/4
650 X=X+XR :: Y=Y+YR :: CALL GCHAR(Y,X,H
):: IF H>95 THEN 700 ELSE IF H=61 THEN 8
00
655 CALL LOCATE(#1,Y*8-7,X*8-7):: ST=ST+
1 :: CALL SOUND(-5,330,0):: GOTO 610
700 FOR I=0 TO 30 STEP 2 :: CALL SOUND(-
100,110,I,220,I):: NEXT I :: CALL SPRITE
(#2,40,16,Y*8-7,X*8-7):: CALL SPRITE(#1,
40,2,(Y*8-7)+8,(X*8-7)+8)
710 FOR I=1 TO 8 :: CALL LOCATE(#2,(Y*8-
7)+I,X*8-7):: NEXT I :: TOT=1 :: CALL DE
LSPRITE(ALL):: M=M-1 :: IF M=0 THEN 3260
0
715 CALL HCHAR(1,3,32,5):: CALL HCHAR(1,
3,40,M-1):: GOTO 545
800 CALL LOCATE(#1,Y*8-7,X*8-7):: CALL S
OUND(-200,380,0):: CALL HCHAR(Y,X,32)::
P=P+100 :: EG=EG+1 :: IF EG=20 THEN 1000
 ELSE 610
1000 CALL HCHAR(2,1,32,736):: DU=1 :: CA
LL DELSPRITE(#1):: GOTO 145
20000 CALL COLOR(1,4,1):: DISPLAY AT(1,7
):"MS SOFT PRESENTS"
20001 DISPLAY AT(3,1):"             #!!!$
#                       !    !
          !    !                       &
   &"
20005 DISPLAY AT(8,1):"  #   %   #   % %  #
  #!!$ #  %  !  !  !  ! !% !  !    !  !
 !!!!  !   ! !&%!  !     !!!!  !  !  !  !
! &!  !     !   !"
20006 DISPLAY AT(12,1):"  &   $   &!!$ $
&  &!!% &  $"
20010 DISPLAY AT(15,1):"     #!!%  #!!%
#!!$  % #$          !  !  !  !  !     !#$
    !!!$  !  !  !       !!       !  %  !!
!!   !      !&%  "
20015 DISPLAY AT(19,1):"     &!!$  &  $
&!!%  & &%"
20020 DISPLAY AT(22,1):" C 1984 BY MICHA
EL SCHNELLER" :: DISPLAY AT(24,1):" PRES
S >>A<< FOR >ANLEITUNG<"
20030 FOR I=3 TO 16 :: CALL COLOR(1,I,1)
:: FOR J=1 TO 20 :: CALL KEY(1,K,S):: IF
 K=18 THEN 20050 ELSE IF K=1 THEN 20100
20035 NEXT J :: NEXT I :: GOTO 20030
20050 CALL CLEAR :: RETURN
20100 CALL CLEAR :: A1$="          ANLEIT
UNG" :: Y=1 :: GOSUB 25000
20105 A1$="1.ABSCHNITT:SIE MUESSEN ANS"
:: Y=4 :: GOSUB 25000 :: A1$="ANDERE END
E DER BURGMAUER" :: Y=6 :: GOSUB 25000
20110 A1$="GELANGEN UND DUERFEN DABEI" :
: Y=8 :: GOSUB 25000 :: A1$="KEINEN PFEI
L BERUEHREN.SIE" :: Y=10 :: GOSUB 25000
20115 A1$="KOENNEN SPRINGEN(FIREBUTTON)"
 :: Y=12 :: GOSUB 25000 :: A1$="UND SICH
 DUCKEN(JOYST. NACH" :: Y=14 :: GOSUB 25
000
20120 A1$="UNTEN.)UND NATUERLICH NACH" :
: Y=16 :: GOSUB 25000 :: A1$="LINKS UND
RECHTS LAUFEN" :: Y=18 :: GOSUB 25000
20125 A1$="!!!!!!!!!!TASTE!!!!!!!!!!!"
 :: Y=21 :: GOSUB 25000
20130 CALL KEY(0,K,S):: IF S<>0 THEN 201
35 ELSE 20130
20135 CALL CLEAR :: A1$="2.ABSCHNITT SIE
 MUESSEN DEN " :: Y=1 :: GOSUB 25000 ::
A1$="TURM HINAUFKLETTERN,DUERFEN" :: Y=3
 :: GOSUB 25000
20140 A1$="SICH DABEI JEDOCH NICHT VON"
:: Y=5 :: GOSUB 25000 :: A1$="EINEM FEUE
RBALL ERSCHLAGEN" :: Y=7 :: GOSUB 25000
20145 A1$="LASSEN." :: Y=9 :: GOSUB 2500
0 :: A1$="!!!!!!!!!!TASTE!!!!!!!!!!!"
:: Y=21 :: GOSUB 25000
20150 CALL KEY(0,K,S):: IF S=0 THEN 2015
0 ELSE CALL CLEAR
20155 A1$="3.ABSCHNITT:SAMMELN SIE ALLE"
 :: Y=1 :: GOSUB 25000 :: A1$="EISBLUMEN
 EIN UND LAUFEN SIE" :: Y=3 :: GOSUB 250
00
20160 A1$="NICHT GEGEN EINE WAND.HABEN"
:: Y=5 :: GOSUB 25000 :: A1$="SIE IHREN
MANN EINMAL IN BE-" :: Y=7 :: GOSUB 2500
0
20165 A1$="WEGUNG VERSETZT,SO LAESST ER"
 :: Y=9 :: GOSUB 25000 :: A1$="SICH DURC
H NICHTS MEHR ALS" :: Y=11 :: GOSUB 2500
0
20170 A1$="EINE WAND(TOEDLICH)STOPPEN" :
: Y=13 :: GOSUB 25000 :: A1$="!!!!!!!!!
!TASTE!!!!!!!!!!!" :: Y=21 :: GOSUB 250
00
20175 CALL KEY(0,K,S):: IF S=0 THEN 2017
5 ELSE CALL CLEAR :: RETURN
25000 FOR I=1 TO LEN(A1$):: CALL HCHAR(Y
,I+2,ASC(SEG$(A1$,I,1))):: CALL SOUND(-1
,-3,1):: NEXT I :: RETURN
31900 DATA -20,-17,-14,-10,1,10,14,17,20
31999 FOR PA=100 TO 104 STEP 4
32000 CALL JOYST(1,X1,Y1):: IF X1+Y1=0 T
HEN CALL MOTION(#1,0,0):: GOTO 32003 ELS
E IF Y1=-4 THEN CALL PATTERN(#1,112):: C
ALL MOTION(#1,0,0):: GOTO 32003
32001 CALL MOTION(#1,0,X1*2):: CALL PATT
ERN(#1,PA):: CALL KEY(1,K,S):: IF K=18 T
HEN 32010
32002 CALL POSITION(#1,Y,X):: IF X<Q THE
N CALL LOCATE(#1,89,Q)ELSE IF X>232 THEN
 32500
32003 CALL COINC(ALL,E):: IF E=-1 THEN G
OSUB 32700
32004 NEXT PA :: GOTO 31999
32005 RETURN
32010 CALL PATTERN(#1,100):: FOR J=-20 T
O 20 STEP 4 :: CALL MOTION(#1,J,X1*3.5):
: CALL COINC(ALL,E):: IF E=-1 THEN GOSUB
 32700 :: GOTO 31999
32015 NEXT J
32025 CALL MOTION(#1,0,0):: CALL POSITIO
N(#1,Y,X):: IF X<Q THEN X=Q
32030 CALL LOCATE(#1,89,X):: GOTO 32000
32050 CALL SPRITE(#1,100,16,89,17,#2,116
,7,89,249,#3,116,7,80,245):: CALL MOTION
(#2,0,-11,#3,0,-19):: GOTO 190
32500 CALL DELSPRITE(#2,#3,#4,#5,#6,#7):
: CALL MOTION(#1,0,0):: FOR V=1 TO 5
32505 FOR SO=0 TO 30 STEP 4 :: CALL SOUN
D(-100,770,SO):: NEXT SO :: FOR SO=0 TO
30 STEP 4 :: CALL SOUND(-100,220,SO):: N
EXT SO :: NEXT V
32510 CALL MOTION(#1,0,-49):: FOR I=1 TO
 10 :: CALL PATTERN(#1,100):: CALL SOUND
(-10,550,2):: CALL PATTERN(#1,104):: CAL
L SOUND(-10,770,2):: NEXT I
32515 CALL MOTION(#1,0,0):: RETURN
32599 !   GAME   OVER
32600 CALL DELSPRITE(ALL):: CALL MAGNIFY
(4):: CALL SPRITE(#1,140,2,49,81,#2,140,
2,49,105,#3,140,2,49,129,#4,140,2,49,155
)
32605 DISPLAY AT(10,11)SIZE(9):"GAME OVE
R"
32610 FOR I=1 TO 4 :: CALL MOTION(#I,0,7
):: NEXT I :: FOR C=4 TO 1 STEP -1 :: FO
```

Scan: S. Höltgen


```
R I=1 TO 155 :: NEXT I :: CALL DELSPRITE
(#C):: NEXT C
32620 DISPLAY AT(24,2)SIZE(26):"PRESS FI
RE BUTTON TO BEGIN"
32630 CALL KEY(1,K,S):: IF K=18 THEN CAL
L CLEAR :: GOTO 10 ELSE 32630
32700 CALL PATTERN(#1,100):: CALL MOTION
(#1,0,0):: FOR SO=1 TO 15 :: CALL SOUND(
-5,110,0):: CALL SOUND(-5,660,0):: NEXT
SO
32705 CALL COLOR(#1,7):: CALL MOTION(#1,
110,0):: FOR C=1 TO 100 :: NEXT C :: CAL
L DELSPRITE(#1):: CALL SCREEN(16):: CALL
 SOUND(50,-6,5)
32710 CALL SCREEN(2):: M=M-1 :: IF M=0 T
HEN 32600 ELSE CALL HCHAR(1,3,32,5):: CA
LL HCHAR(1,3,40,M-1)
32720 CALL SPRITE(#8,140,2,89,42,#9,140,
2,89,70):: FOR I=1 TO 3 :: NEXT I
32725 CALL COINC(ALL,E):: IF E=-1 THEN 3
2725
32730 CALL DELSPRITE(#8,#9):: CALL SPRIT
E(#1,100,16,89,34):: RETURN
```

Schneider: Kamikaze

```
10 ON ERROR GOTO 790:DEFINT A-Z:D
IM A(5)
20 ST=1:HTANK=20:L=1
30 MODE 1:INK 0,2:INK 1,13:INK 2,
1:INK 3,26
40 BORDER 2:PAPER 0:PEN 1
50 SYMBOL AFTER 200
60 SYMBOL 228,255,255,255,255,255
,255,255,255 'REV.LEERZEICHEN
70 SYMBOL 226,66,38,30,30,60,120,
24,24:SYMBOL 227,34,127,42,8,8,8,
8,28 'PARABOLANTENNE
80 SYMBOL 222,0,0,0,0,0,60,126,36
 'BOMBE 1
90 SYMBOL 223,0,0,0,0,0,36,126,60
 'BOMBE 2
100 SYMBOL 235,255,255,146,146,14
6,255,255,255:SYMBOL 236,255,255,
73,73,73,255,255,255 'STOCKWERK 1

110 SYMBOL 237,128,156,170,190,17
0,190,128,255:SYMBOL 238,1,57,85,
125,85,125,1,255 'STOCKWERK 2
120 SYMBOL 239,255,201,255,201,25
5,201,255,255:SYMBOL 240,255,147,
255,147,255,147,255,255 'STOCKWER
K 3
130 SYMBOL 224,0,0,0,30,127,255,3
1,0:SYMBOL 225,0,0,3,7,255,253,19
2,0 'FLUGZEUG 1
140 SYMBOL 229,32,112,112,40,6,2,
0,0:SYMBOL 230,4,24,0,2,12,104,19
2,96:SYMBOL 231,8,24,8,0,19,119,1
27,255:SYMBOL 232,127,127,51,16,0
,32,96,32:SYMBOL 233,4,14,6,26,16
,32,128,192:SYMBOL 234,148,8,128,
0,100,39,22,2 'EXPLOSION
150 SYMBOL 241,3,15,31,63,99,85,1
93,255:SYMBOL 242,192,240,248,252
,198,170,131,255 'DACH 1
160 SYMBOL 243,0,0,0,0,3,15,63,25
5:SYMBOL 244,0,0,32,240,248,252,2
54,255 'DACH 2
170 SYMBOL 245,0,0,192,224,255,12
7,3,0:SYMBOL 246,0,0,0,120,254,25
5,248 'FLUGZEUG 2
180 '
190 GOSUB 660 'TITELBILD
200 COUNT=5:DIR=1:X=2:Y=3:FLAG=0:
IF HTANK>0 THEN TANK=HTANK ELSE T
ANK=0
210 PEN 2:CLS:GOSUB 490
220 FLAG1=0:GOSUB 380 'STADT AUFB
AUEN
230 IF JOY(0)=1 OR INKEY(0)=0 THE
N IF Y>3 AND TANK>0 THEN TANK=TAN
K-1:Y=Y-1:GOSUB 500
240 IF (JOY(0)=2 OR INKEY(2)=0) A
ND Y<24 THEN Y=Y+1
250 IF TEST(X*16-16*(1-DIR),400-Y
*16)<>0 THEN 610
260 LOCATE X,Y:PEN 2:IF DIR=1 THE
N PRINT CHR$(245)CHR$(246) ELSE P
RINT CHR$(224)CHR$(225)
270 IF (JOY(0)=16 OR INKEY(47)=0)
 AND FLAG=0 THEN FLAG=1:XB=X:YB=Y
280 FOR T=0 TO 150-L*50:NEXT
290 IF FLAG=1 AND (TEST (XB*16-13
,400-YB*16-12) <>0 OR TEST(XB*16-
16,400-YB*16-16)<>0 ) THEN GOSUB
480:COUNT=COUNT-1 ELSE FOR T=1 TO
 100:NEXT
300 IF FLAG=1 THEN LOCATE XB,YB:P
RINT" ":YB=YB+1:LOCATE XB,YB:PEN
3:B=(B+1)AND 1:PRINT CHR$(222+B):
PEN 2
310 IF (COUNT<0 OR YB>23) AND FLA
G=1 THEN FLAG=0:COUNT=5:LOCATE XB
,YB:PRINT " "
320 LOCATE X,Y:PRINT "  "
```

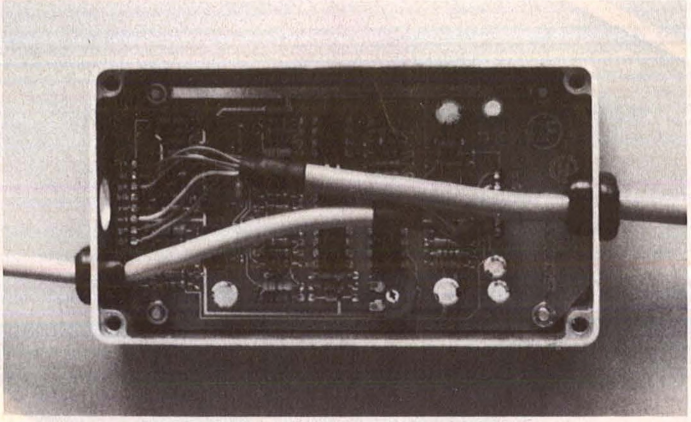
Scan: S. Höltgen

Unverwüstlich – der TI

Immer wieder kommen Neuheiten für den TI99/4A auf den Markt. Die treuen Fans des Oldtimers wissen es zu schätzen. HC stellt die interessantesten Neuheiten vor.

Der Printer-Plotter PP-A4 läßt Papierbreiten von 100 bis 216 mm zu. Das Papier wird in Längsrichtung bewegt und kann sowohl in Rollen als auch in Einzelblättern verarbeitet werden. Somit ist es auch möglich, DIN-A4-Blätter zu verwenden. Vom Programm können vier Kugelschreiberminen angesteuert werden. Über DIP-Schalter sind vier Zeichensätze für den Textmodus auswählbar. 15 Linienarten werden bereitgestellt. Die Zeichengeschwindigkeit beträgt ca. 60 mm pro Sekunde. Der Printer-Plotter ist bei Alltronic und Mechatronic erhältlich. Sein Preis ist ungefähr 700

Linientreu *erscheint auf dem Bildschirm, was man mit dem Zeichenarm des Grafiktabletts auf einer Vorlage abfährt*



Monitoranschluß *ist mit dem RGB-Modul auch endlich beim TI möglich*

Der richtige Dreh *genügt, und schon ist beim Modul-Expander eine andere Cartridge zugeschaltet*

Mark, das Anschlußkabel für die 32-K-RAM-Erweiterung kostet zusätzlich ca. 70 Mark.

Auf einer Arbeitsfläche von 22 mal 19 Zentimeter kann beim Grafik-Tablett Super Sketch ein Zeichenarm frei bewegt werden, um die Konturen einer eingespannten Vorlage nachzuziehen. Zur weiteren Gestaltung des Ergebnisses auf dem Bildschirm stehen 16 Farben, verschiedene Schriftbreiten und -muster zur Verfügung. Radieren, Farbwechsel und Ausmalen von Flächen ist möglich. Das Zeichnen von Kreisen, Ellipsen und Rechtecken wird unterstützt, Texte können eingeblendet werden. Der Anschluß erfolgt über ein Kabel, dessen Stecker in einer ROM-Cartridge besteht. Radix-Bürotechnik und Computer-Hüsli bieten das Grafik-Tablett zum Preis von ca. 250 Mark an.

Ein RGB-Modulator von Computer-Hüsli für ca. 180 Mark macht nun endlich auch den Anschluß eines RGB-Monitors an den TI99/4A möglich.

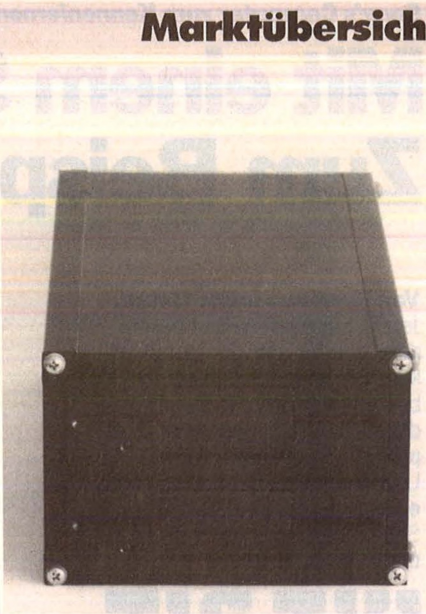
Ebenfalls von Computer-Hüsli ist ein AD/DA-Wandler für rund 200 Mark. An das Modul können bis zu acht analoge Geräte angeschlossen werden. Voraussetzung für seinen Einsatz sind entweder das Extended-BASIC-Modul oder das Editor-Assembler-Modul.

Beim Modul-Expander läßt sich mit einem Drehschalter bequem zwischen acht verschiedenen ROM-Cartridges wählen.

Der Akustikkoppler Dataphon S21D ermöglicht zusammen mit dem Terminal-Emulator II die Kontaktaufnahme zu anderen Computern über das Telefonnetz. Sowohl Modul-Expander (etwa 230 Mark) als Akustikkoppler (mit Anschluß-

Scan: S. Höltgen

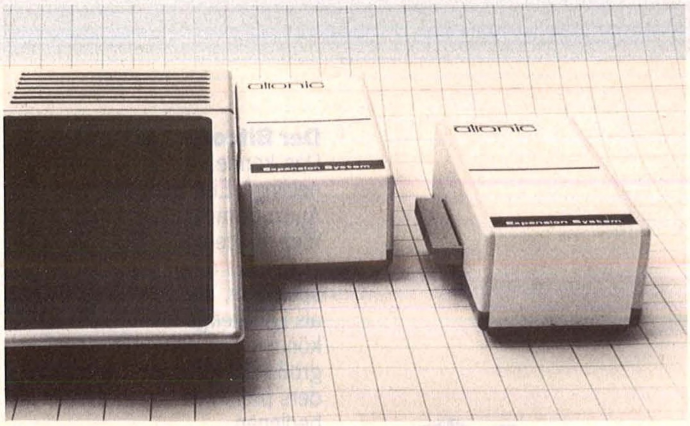
Kontakte *knüpft der TI zu anderen Computern mit dem Akustikkoppler*



Ihre schlanke Linie *macht die Slimline-Laufwerke attraktiv*

Betriebssystemerweiterung *TI99-PP ohne separate Stromversorgung*

Das Peripheriesystem *CPS99 enthält unter anderem eine 32-KByte-Speichererweiterung, mehrere Schnittstellen und einen Disk-Controller*



Speichererweiterung *bis 128 KByte sind erhältlich*

kabel und Emulator ca. 400 Mark) werden von Computer-Hüsli angeboten.

Disketten-Laufwerke mit halber Bauhöhe („slimline") für 5,25"-Disketten, die doppelseitig bespielbar sind und doppelte Dichte haben, sind sowohl für die Peripherie-Box (mit entsprechendem Einbausatz) als auch in eigenem Gehäuse von Alltronic und von Computer-Hüsli lieferbar. Zum Betrieb ist jeweils ein Disketten-Controller erforderlich.

Ein Disketten-Laufwerk für 2,8"-Disketten zum Direktanschluß an den TI99/4A kommt ebenfalls von Alltronic.

Die Betriebssystemerweiterung TI99-PP von Penta Electronics und Computer-Hüsli wird seitlich an den TI99/4A gesteckt und benötigt keine separate Stromversorgung. Es sind zum Beispiel möglich: Peeken und Poken in allen Speicherbereichen, Ausgabe von Hex-Dumps auf Bildschirm oder Drukker, Betreiben des Printer-Ports als Centronics-Schnittstelle.

Unter der Bezeichnung CPS99 läuft ein komplettes Peripheriesystem von Atronic. Es wird seitlich an die Konsole angesteckt und enthält eine 32-KByte-Speichererweiterung, zwei RS232-Schnittstellen, eine Centronics-Schnittstelle, einen Disk-Controller zum Betrieb von maximal zwei 5,25"-Laufwerken, wahlweise einseitig oder doppelseitig, mit einfacher oder doppelter Dichte, und einen Diskmanager auf Diskette.

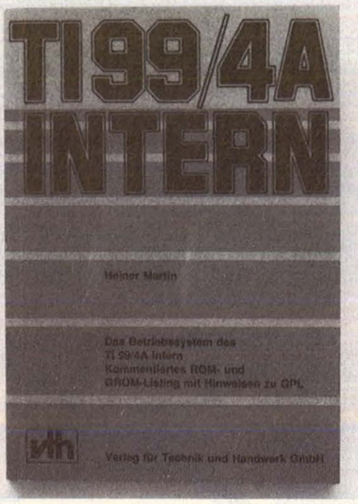
Von Radix Bürotechnik kommt DOS-80 auf zwei Eproms zum Umbau des TI-Disk-Controllers zwecks Anschluß von wahlweise 40- oder 80-Spur-Laufwerken, einseitig oder doppelseitig. Die Laufwerktypen können auch gemischt eingesetzt werden.

32-KByte-Speichererweiterungen zum seitlichen Anschluß an die TI-Konsole sind mit und ohne zusätzliche Centronics-Schnittstelle lieferbar, in Versionen mit statischen oder dynamischen RAM-Bausteinen. Sogar ausgebaut zu einer 128-KByte-Erweiterung ist sie bei Mechtronic und Alltronic erhältlich.

Mit Extended BASIC II plus

Scan: S. Höltgen

Ein kompletter Grafikbefehlssatz *(über 35 neue Befehle) und viele andere Erweiterungen machen das Programmieren in BASIC mit Hilfe das Extended BASIC II plus wesentlich komfortabler*



Letzte Geheimnisse *lassen sich mit den ROM-Listings aufklären*

Know-how *zur Datenfernübertragung vermittelt ein Handbuch*

wurde das Extended BASIC noch erheblich ergänzt (Mechatronic, Alltronic). Es umfaßt einen kompletten Grafik-Befehlssatz (über 35 neue Befehle), der allerdings nur mit der Speichererweiterung nutzbar ist. Die Grafik ist sehr schnell und erlaubt den Zugriff auf den ganzen Bildschirm, auch wenn im Einzelnen nur ein Fenster von 128 × 120 Punkten angesteuert werden kann. Das Fenster kann aber mehrfach auf dem Bildschirm abgebildet werden. Selbstverständlich ist jeder Punkt in diesem Fenster einzeln ansteuerbar. Auch ohne die Speichererweiterung stehen neue Befehle zur Verfügung. An erster Stelle ist hier wohl eine Hardcopy-Routine für Matrixdrukker zu nennen, die natürlich auch mit der Grafik nutzbar ist. VPEEK und VPOKE lassen einen direkten Zugriff auf das RAM des Video-Controllers zu (Preis ca. 300 Mark). Besitzer des bisherigen Lizenznachbaues können ihr Extended-BASIC für ca. 100 Mark umrüsten.

Eine interessante Erweiterung zum Editor/Assembler-Modul von Texas Instruments ist der TI-99/4A-Monitor von Jens Sundermann zum Preis von ca. 55 Mark für die Diskette mit ausführlicher Beschreibung. Er wird wie ein übliches Maschinensprache-Programm geladen und übernimmt fortan alle Funktionen des Editors und des Assemblers. Der große Vorteil liegt darin, daß man nicht fortlaufend ,,Diskjockey" spielen muß, denn das ständige Laden von Editor und Assembler entfällt. Ferner gibt es nicht mehr die Unterscheidung zwischen Source-File und Object-File. Auch die Assembler Optionen gehören der Vergangenheit an.

Der Monitor arbeitet mit einem Einzeilenassembler, welcher sofort nach der Eingabe des Mnemonic-Codes assembliert. Der volle Befehlssatz aller TMS-9900-Mnemonics ist in allen Formaten verfügbar. Wie schon gesagt, wird nur mit einem File gearbeitet, da der Monitor einen Disassembler besitzt, der das erstellte Object-File in seiner exakten Mnemonic-Form wieder rekonstruiert. Eine nützliche Ergänzung zum Editor-Assembler ist auch der Single-Step-Mode, der es erlaubt, das Object-File Zeile für Zeile abzuarbeiten, und somit ein Auffinden von Fehlern wesentlich erleichtert.

Weitere Vorteile des Monitors:
- Umwandeln von Dezimal- in Hexadezimalzahlen
- Umwandeln von Hexadezimal- in Dezimalzahlen
- Hexadezimale Arithmetik
- Breakpoints setzen und ausdrucken, wobei bis zu vier Breakpoints gesetzt werden können
- Auffinden von 8/16-bit-Zahlen
- Verschieben von Speicherbereichen mit oder ohne Korrektur der Sprungadressen
- Ausdrucken und Laden von Program-Counter, Workspace-pointer und Statusregister
- Laden der Workspace-Register
- Ausdrucken eines Speicherbereiches

Der Monitor kann sowohl eigenerstellte Object-Code-Programme als auch mittels TI-Assembler erstellte Object-Code-Programme lesen und verarbeiten, so daß bisher erstellte Maschinenprogramme auch mit dem Monitor weiterverarbeitet werden können. Mit dem Monitor befindet sich ein Programm namens P-LIST auf der Diskette, welches den Ausdruck des Diskettenkataloges ermöglicht und mit dem Monitor gleichzeitig geladen werden kann.

Erforderliche Gerätekonfiguration zum Betrieb des Monitors:
- TI99/4A-Konsole
- Editor/Assembler-Modul
- 32-KByte-RAM-Expansion
- Mindestens ein Diskettenlaufwerk

Das Editor/Assembler-Modul ist erforderlich, da sonst die Möglichkeiten des Monitors nicht vollständig ausgenutzt werden können. Es ist jedoch eine Version des Monitors in Vorbereitung, die Assembler-Programmierung unter Verwendung nur des Extended-BASIC-Moduls erlaubt.

Die Programmiersprache Forth gibt es in zwei Versionen für den TI99/4A: als Diskette aus den USA (bei vielen Fachhändlern) und als Modul (in Deutschland erstellt und nicht identisch mit der Diskettenversion, bei Firma Schuh).

Einfache Tabellenkalkulation für den Heimgebrauch oder eine kleine Firma ist mit TI-Calc möglich (im Fachhandel).

Die Datenfernübertragung prägnant und leicht verständlich zu beschreiben, wurde mit dem ,,Handbuch zur Datenfernübertragung" von Klaus Weidemann versucht (Radix Bürotechnik).

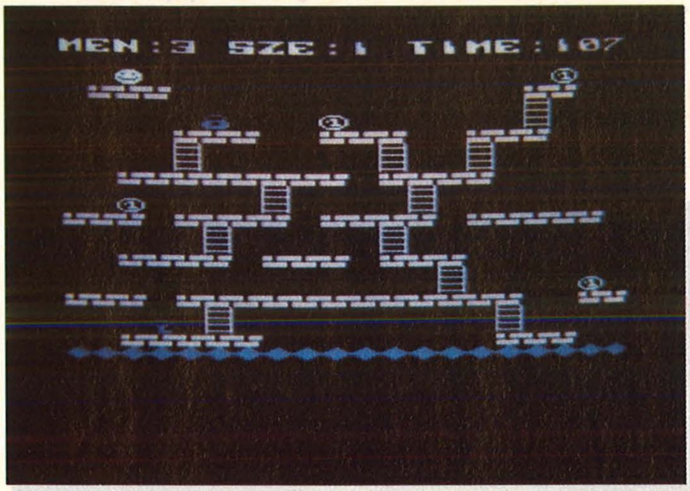
Einen Rechner ganz verstanden hat man immer erst dann, wenn man auch die letzten Ecken seines Betriebssystems beleuchtet hat. Die Listings des ROM und der GROM's finden sich mit einer ausführlichen Analyse im Buch ,,TI99/4a intern" von Heiner Martin im Verlag für Technik und Handwerk GmbH (bei Computer-Hüsli ca. 40 Mark).

-br

Scan: S. Höltgen

Zum Eintippen

Der goldene Keller



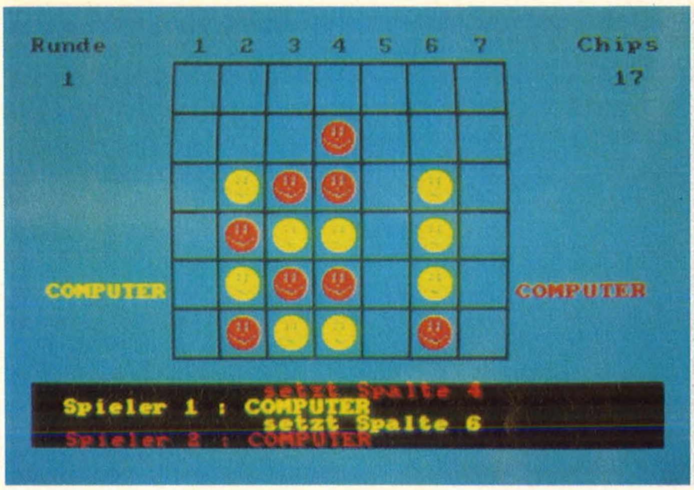
Nicht einfach ist es, die Goldstücke im Labyrinth einzusammeln, ohne dabei die Spinne zu berühren ***(Atari 800XL)*** *Listing auf Seite 51*

Mini-Mühle

*Der **TI-99/4A** + Extended-BASIC wird von Spiel zu Spiel intelligenter und somit zur Herausforderung für den Spieler* *Listing auf Seite 53*

Vier gewinnt

Das Strategiespiel mit hoher Spielstärke und der Möglichkeit, bereits gesetzte Steine zurückzunehmen ***(Schneider CPC 464)*** *Listing auf Seite 58*



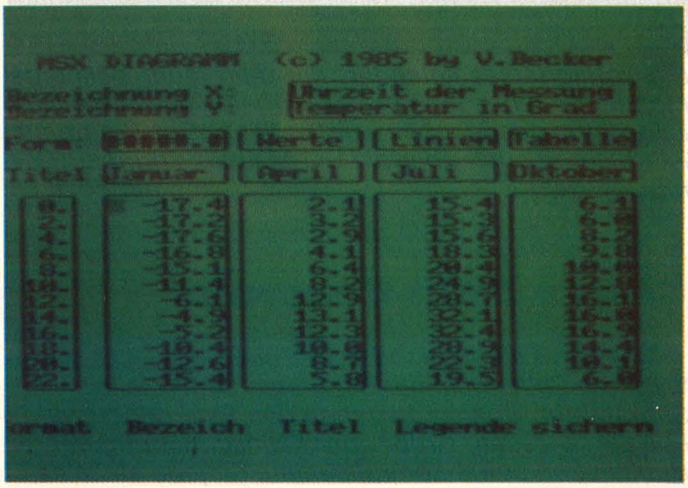
Feld-Billard

*Zielfeld, Abschußpunkt und Abschußwinkel bestimmen — und ab geht die Kugel. Größe des Zielfeldes und die Anzahl der gespielten Banden erhöhen das Trefferrisiko (alle **Apple II**)* *Listing auf Seite 69.*

INSTR-Funktion

*Dieser Algorithmus findet ein gesuchtes Wort auch in längeren Strings. Dabei dürfen Fehler wie falsche, fehlende sowie überflüssige Zeichen auftreten **(Colour Genie)*** *Listing auf Seite 73.*

Diagramm



Scan: S. Höltgen


```
WB 2075 DATA ####################
WF 2076 DATA ####################
SC 2077 DATA #"""########"""#####
TR 2078 DATA #!############!#####
ZH 2079 DATA #!##a####a####!#####
MV 2080 DATA #""""####"""##""""##
SO 2081 DATA ##!############!####
SS 2082 DATA ##!############!####
LR 2083 DATA ##""""##"""""""""###
UO 2084 DATA ########!###########
US 2085 DATA ########!###########
PF 2086 DATA ""######"""""####""#
UU 2087 DATA ###########!########
XM 2088 DATA ###▒#######!#####▒##
OS 2089 DATA #"""######""""#"""##
SZ 2090 DATA #!########!#########
TD 2091 DATA #!########!#########
NF 2092 DATA #"""""##"""""""#####
RL 2093 DATA ####!###!#####!#####
UX 2094 DATA ####!#a#!#####!#a###
MJ 2095 DATA ###""""""####""""""#
NT 2096 DATA ((((((((((((((((((((
UO 2100 DATA ####################
US 2101 DATA ####################
MY 2102 DATA """"#""""""""#######
TK 2103 DATA ##########!#########
BE 2104 DATA ##########!###a#####
NV 2105 DATA ###"""#""""#"""#####
UC 2106 DATA #######!############
SI 2107 DATA #######!##a#########
IG 2108 DATA ((((#""""#"""#""""##
SI 2109 DATA ########!########!##
WB 2110 DATA ##a#####!########!##
MJ 2111 DATA ##""""#""""#####"""#
RN 2112 DATA #######!########!###
IR 2113 DATA #######!▒#######!###
EN 2114 DATA ""##""""""((((#""""#
QT 2115 DATA ####!####!#####!####
TN 2116 DATA #a##!####!#####!####
MT 2117 DATA #""""###"""""""#####
SN 2118 DATA ########!#####!#####
IU 2119 DATA ########!#####!###▒#
LE 2120 DATA ####"""""#####""""""
MG 2121 DATA ((((((((((((((((((((
EV 2122 DATA -1
FH 2125 DATA -1
JT 6000 CH=PEEK(106)-8:CHS=CH*256:FOR LOO
   P=CHS TO CHS+256:POKE LOOP,0:POKE LOOP
   +256,0:POKE LOOP+512,0
BM 6001 POKE LOOP+768,0:NEXT LOOP:RESTORE
    6005
IA 6002 READ A:IF A=-1 THEN RETURN
KP 6003 FOR LOOP=0 TO 7:READ B:POKE CHS+A
   *8+LOOP,B:NEXT LOOP:GOTO 6002
NI 6005 DATA 5,60,66,66,126,231,231,255,1
   26,2,0,239,239,239,0,254,254,254
EM 6006 DATA 1,129,129,255,129,129,129,25
   5,129,33,60,66,153,169,137,157,66,60,6
   ,60,66,66,60,24,24,30,30
YU 6007 DATA 7,60,126,219,255,189,195,126
   ,60,8,24,60,126,255,255,126,60,24
IL 6008 DATA 37,254,198,192,248,224,230,2
   54,0,41,48,48,48,56,56,56,56,0,45,198,
   238,254,246,230,230,230,0
MI 6009 DATA 46,198,230,246,254,238,230,2
   30,0,51,254,198,192,254,14,206,254,0,5
   8,254,12,24,48,112,246,254,0
WT 6010 DATA 52,254,48,48,56,56,56,56,0,2
   6,0,24,24,0,0,24,24,0
XH 6011 DATA 16,124,206,222,246,230,230,1
   24,0,17,48,48,48,56,56,56,56,0,18,252,
   206,216,48,96,224,254,0
JC 6012 DATA 19,254,198,6,62,14,206,254,0
   ,20,198,198,198,254,14,14,14,0,21,254,
   192,192,254,6,230,254,0
UD 6013 DATA 22,254,198,192,254,230,230,2
   54,0,23,254,204,24,48,96,192,192,0,24,
   124,108,108,254,230,230,254,0
IZ 6014 DATA 25,254,198,198,254,6,230,254
   ,0,9,24,24,24,66,60,90,90,153,-1
TO 7000 DIM C$(2)
NE 7010 RESTORE 7500
DK 7015 GRAPHICS 1:SETCOLOR 2,0,0
BU 7020 READ C$:IF C$="-1" THEN ? #6:? #6
   :? #6:GOTO 7020
FK 7025 IF C$="-2" THEN GOTO 7600
PX 7030 ? #6;C$;:SOUND 0,200,14,10:FOR T=
   0 TO 5:NEXT T:SOUND 0,0,0,0:GOTO 7020
SO 7500 DATA , , , ,O,R,R,E,N, ,F,I,E,D,L
   ,E,R,-1
ZI 7501 DATA , , , , , ,P,R,E,S,E,N,T,S,-
   1,-1,-1
LF 7502 DATA , , ,t,h,e, ,g,o,l,d,e,n, ,c
   ,e,l,l,a,r,-2
DH 7600 ? "Einen Moment, bitte..."
AF 7601 ? :? "Copyright by Orren Fiedler
   1985"
AU 7610 RETURN
```

Mini-Mühle

MINI-MÜHLE ist ein altes Brettspiel, ähnlich dem TIC-TAC-TOE. Das Spielfeld zeigt 3 × 3 Felder, und jeder der beiden Spieler hat drei Steine, die mit 1, 2 und 3 numeriert sind. Die Spieler sind abwechselnd am Zug und springen jeweils mit einem eigenen Stein auf ein freies Feld. Dabei ist immer der Stein zu ziehen, der in der Reihenfolge 1, 2, 3 als nächster an der Reihe ist. Ziel des Spiels ist es, seine drei Steine in eine gerade Reihe zu bringen, senkrecht, waagerecht oder diagonal. Dieses Spiel ist eigentlich eher langweilig, weil es keiner gewinnen kann, solange beide Spieler keinen Fehler machen. Der Computer macht aber Fehler, weil er anfangs nur die Spielregel kennt und nur eine sehr einfache Taktik hat. Wenn er eine eigene Dreierreihe vollenden kann, dann macht er das, wenn nicht, schaut er nach, ob der Spieler im nächsten Zug eine Dreierreihe vollenden kann. Diese verhindert er dann. Trifft beides nicht zu, setzt er seinen Stein auf das erste freie Feld. So ist es also zunächst nicht schwer, den Computer zu besiegen. Allerdings „lernt" der Computer aus seinen Fehlern, und nach mehreren Partien werden Sie nicht mehr so schnell gewinnen, vielleicht verlieren Sie sogar. Da man das Spiel durch Wiederholung der immer gleichen Spielzüge bis in Ewigkeit fortsetzen kann, macht in einer solchen Situation der Computer absichtlich einen anderen Zug, der höchstwahrscheinlich Sie zum Gewinn führt. Nach maximal 63 Zügen gilt das Spiel als unentschieden.

Damit Sie nicht nach jedem Einschalten des Computers wieder einen ganz „dummen" Gegner haben, können Sie die Computererfahrungen aufzeichnen und beim nächsten Mal wieder einlesen. So wird der Com-

Scan: S. Höltgen

puter mit der Zeit immer intelligenter. Auf der Titelseite werden Sie gefragt, ob der Computer schon gelernt haben soll. Drückt man „J", so wird man nach dem Eingabegerät gefragt. Es ist der Kassettenrekorder CS1 möglich oder die Diskdrives DSK1 oder DSK2. Sind die Daten auf Diskette, wird man noch nach dem Dateinamen gefragt. Wird die Datei nicht gefunden, ist der Rekorder nicht an der richtigen Stelle oder gar nicht angeschlossen oder der Dateiname wird auf der Diskette nicht gefunden. Dann wird das Spielfeld gemalt: Neun schwarze Felder und die Erläuterungen zeigen an, daß Sie mit gelben Steinen spielen, der Computer mit roten. Wollen Sie eine Partie abbrechen, müssen Sie das „et"-Zeichen drücken. Die ersten Steine, die Sie setzen, werden nur per Text angezeigt. Sie tippen die Feldkoordinaten, zum Beispiel A1, und sofort flackert auf diesem Feld eine schwarze 1 auf gelbem Grund. Mit 0 können Sie diesen Zug zurücknehmen, mit ENTER bestätigen. Dann folgt der Computer. Damit Sie sehen können, wohin er setzt, flackert auch sein Stein so lange auf dem Feld, bis Sie ENTER drücken. Wenn alle Steine auf dem Brett sind, wird durch inverse Farbgebung der Stein angezeigt, der gezogen werden muß. Unerlaubte Züge werden nicht angenommen, der Fehler angezeigt. Sobald eine Partei gewonnen hat, blinkt die Siegesreihe, man kann eine neue Partie beginnen oder das Spiel beenden. Will man aufhören, kann man die Erfahrungen des Computers aufzeichnen. Man wird wieder nach dem Speichergerät gefragt wie beim Einlesen.

Bernhard Decker

Zeilen	Kommentar
10– 40	Initialisierung
50– 140	Einlesen Computer-Erfahrungen
150– 250	Sonderzeichen definieren
260– 350	Aufbau Bildschirm
370– 440	Initialisierung (Fortsetzung)
460– 800	Spielerzug
720	Zugzurücknahme
750	Update Spielbrettarrays
780	Registrierung Spielzug
820–1060	Computerzug
900– 950	Auswahl des Computersteinzielfeld
1040–1050	Registrierung des Computerzugs. Wenn Zug zwangsläufig, dann Code um neun erhöht
1080–1240	Update der gespeicherten Partien, wenn Spieler gewonnen hat
1240–1330	Update der gespeicherten Partien, wenn Computer gewonnen hat
1340–1540	Endbehandlung, Erfahrungen aufzeichnen
1560–3220	Subroutinen
1560–1610	Berechnung der Zugsteinnummer
1630–1930	Je nach Lage der Steine bekommen die Reihen R() einen Wert. Die Einzelfelder bekommen die Wertsumme aller Reihen, zu denen sie gehören
1950–1970	Wenn Computer ohne Erfahrung → Zufallszug
1990–2030	Zuerst wird gleiche Computergewinnpartie gesucht
2040–2080	dann gleiche Computerverlustpartie
2080–2110	gespielte Partie wird mit Altpartie verglichen
2120–2160	errechnet nächstes freies Feld
2180–2210	erstes freies Feld wird gesucht
2230–2240	Zufallszug
2260–2300	Check, ob stete Zugwiederholung
2320–2380	Sucht alte Gewinnpartie zum Nachspielen
2400–2470	Neuspiel wird um Zwangszüge gekürzt
2490–2550	Verlustpartien nach Länge sortieren
2570–2600	Gewinnpartien nach Länge sortieren
2620–2650	Löcher in Gewinnpartieliste aussortieren

Zeilen	Kommentar
2670–2700	Alte Verlustpartien mit neuer vergleichen und bei Gleichheit löschen
2720–2760	Alte Verlustpartien prüfen, ob Spieler neue Partie durch Fehler verloren hat
2780–2810	Check, ob Spieler genau gleich gespielt hat
2830–2860	Gelöschte Partien (falsche Erfahrungen) und Löcher schließen
2880–2980	Endetest
3000–3010	Schreibfläche löschen
3030	Erfahrungen auf Speichergerät schreiben
3050	Erfahrungen von Speichergerät lesen
3070–3120	Speichergeräteauswahl
3140–3220	Je nachdem, wie der erste Stein gesetzt wird, dreht der Computer für sich das Spielfeld, um gleiche Spiele bei verdrehtem Spielbrett zu erkennen
3230–3260	Die neun Spielfelder werden von oben zeilenweise mit Buchstaben (A bis I) und Zahlen (1 bis 9) numeriert und für die vier möglichen Lagen des Bretts verändert. Diese Codes sind in den DATAS gespeichert.

Programmstruktur

Scan: S. Höltgen

Variable	Bedeutung
A	Steinkoordinate waagerecht (Spielbrettzeile)
A$	Zwischenspeicher
AC()	Spielbrettzeile von Computerstein 1 bis 3
AS()	Spielbrettzeile von Spielerstein 1 bis 3
AS$	Speicher eines ganzen Spiels mit besonderer Markierung der Computerzwangszüge
B	Steinkoordinate senkrecht
B$	Zwischenspeicher
BC()	Spielbrettspalte von Computerstein 1 bis 3
BG	Flag für Startspieler (1 = Spieler, 2 = Computer)
BS()	Spielbrettspalte von Spielerstein 1 bis 3
BS$	Speicher eines ganzen Spiels
C	Zwischenspeicher bei Endetest, 1 = kein Ende, 2 = Computer gewinnt, 3 = Spieler gewinnt, 10 = abbrechen; Zähler Computersteine bei Zielfeldberechnung
C1	Zähler Spielersteine bei Zielfeldberechnung
D	Steinnummer, die Gegner als nächste ziehen muß
D$	Zwischenspeicher
D5	Flag, 1 = 1. Stein im Mittelfeld
E$	Dateiname
F(,)	Spielbrett mit allen Steinen
F1(,)	Spielbrett, Wert als Computersteinzielfeld
F2(,)	Spielbrett ohne Zugstein
G	Flag, 1 = Verlustpartie, 2 = Gewinnpartie
H	Status in Keyroutine und Zwischenspeicher
I	Schleifenzähler
J	Schleifenzähler und Zwischenspeicher
L1,L2	Speicher für Stringlängen
M(,)	Anzahl gelernter Partien M(BG/G)
N	Flag, 1 = alte Partie wird anders fortgesetzt
N$	Speicher für neue Partie
N1	Flag, 1 = alte Partie wird früher als erst beim letzten Zug anders gespielt
NR	Halbzugzähler zur Ermittlung der Steinnummer
P	Nummer des zu spielenden Steins
R()	Werte der acht möglichen Reihen
S	Flag, 1 = Stein muß auf errechnetes Feld gesetzt werden, um Gewinn zu erreichen oder Verlust zu vermeiden (Zwangszug)
S$(,,)	Partienspeicher, S$ (BG,G,Anzahl)
X	Zugzähler
Y	Flag, 1 = Spieler verliert durch Fehler
Z()	Zahlencode für die neun Felder des eventuell verdrehten Spielbretts
Z$(,)	Buchstabencode des Feldes mit den Koordinaten A und B zur Partiespeicherung

Liste der Variablen

```
10 DIM R(8),F(3,3),F1(3,3),F2(3,3),AC(3)
,BC(3),AS(3),BS(3),Z$(3,3),S$(2,2,40),M(
2,2)
20 RANDOMIZE
30 CALL COLOR(1,2,16):: CALL CLEAR :: CA
LL SCREEN(3)
40 FOR I=0 TO 8 :: CALL COLOR(I,2,16)::
NEXT I
50 PRINT "MINI-MUEHLE": : : : : :"COPYRI
GHT 1983": :"BY": :"BERNHARD DECKER": :
: :"SOLL DER COMPUTER SCHON GE- LERNT HA
BEN? J": : : :
60 ACCEPT AT(20,14)SIZE(-1)VALIDATE("JN"
):A$ :: IF A$="N" THEN 150
70 H=0 :: GOSUB 3070 :: OPEN #1:E$,INPUT
 ,FIXED 80,INTERNAL
80 IF H<>0 THEN 70 ELSE CALL CLEAR :: PR
INT "LERNSPIEL": : : :
90 INPUT #1:M(1,1),M(1,2),M(2,1),M(2,2)
100 PRINT "COMPUTER-ERFAHRUNGEN": :"GEWI
NN  =";M(1,2)+M(2,2):"VERLUST =";M(1,1)+
M(2,1): : :
110 A,B=1 :: GOSUB 3050
120 B=2 :: GOSUB 3050
130 A=2 :: B=1 :: GOSUB 3050
140 B=2 :: GOSUB 3050 :: CLOSE #1
150 CALL CHAR(104,"00FFFFFFFFFFFF007E7E7
E7E7E7E7E7E")
160 CALL CHAR(96,"00"&RPT$("7F",15)&"00"
&RPT$("F",30))
170 CALL CHAR(112,"00FFFFFFFFFFFF007E7E7
E7E7E7E7E7E")
180 CALL CHAR(120,"007F7F7F7E7C797B7F7F7
F7F7F78787F00FF9F1F1F9F9F9F9F9F9F9F9F070
7FF")
190 CALL CHAR(124,"007F7E7C78797B7F7F7F7
E7C79787877F00FF1F0FC7E7E7CF9F3F7FFFFF070
7FF")
200 CALL CHAR(128,"007F7E7C797B7F7F7F7F7
F7B797C7E7F00FF3F0FC7E7E7C71FCFE7E7C70F3
FFF")
210 CALL CHAR(132,RPT$("F",16),136,RPT$(
"F",16),137,"")
220 CALL CHAR(138,"80C0E0F0F8FCFEFF",40,
"FF7F3F1F0F070301",100,"80C0E0F0F8FCFEFF
")
230 CALL SPRITE(#1,120,11,193,1,#2,124,1
1,193,1,#3,128,11,193,1)
240 CALL SPRITE(#4,120,7,193,1,#5,124,7,
193,1,#6,128,7,193,1)
250 CALL COLOR(2,13,16,9,4,16,10,11,16,1
1,7,16,13,2,16,14,13,4)
260 CALL MAGNIFY(4):: CALL CLEAR
270 DISPLAY AT(1,1):"MINI-MUEHLE      IHR
E FARBE:":TAB(16);"COMPT.FARBE:":TAB(18)
;"ABBRECHEN:"
280 CALL HCHAR(1,31,104):: CALL HCHAR(2,
31,112):: CALL HCHAR(3,31,64)
290 DISPLAY AT(5,4):"1    2    3": : :"A
": : : : :"B": : : : :"C"
300 FOR I=6 TO 19 :: CALL HCHAR(I,4,132,
14):: NEXT I
310 FOR I=10 TO 20 STEP 5
320 CALL HCHAR(I,5,136,13):: CALL VCHAR(
7,-2+I,137,13):: NEXT I
330 FOR I=1 TO 3 :: FOR J=1 TO 3
340 CALL HCHAR(1+5*I,3+5*J,100):: CALL H
CHAR(5+5*I,-1+5*J,40):: CALL HCHAR(5+5*I
,3+5*J,138)
350 NEXT J :: NEXT I
360 REM
370 FOR I=1 TO 3 :: FOR J=1 TO 3 :: F(I,
J),F1(I,J),F2(I,J)=0 :: NEXT J :: NEXT I
380 N$,BS$,AS$=""
390 FOR I=1 TO 6 :: CALL LOCATE(#I,193,1
```

Scan: S. Höltgen


```
):: NEXT I
400 FOR I=1 TO 3 :: AS(I),BS(I),AC(I),BC
(I)=0 :: NEXT I
410 NR=.5 :: C,X=0 :: CALL HCHAR(21,1,32
,128)
420 DISPLAY AT(22,1):"WOLLEN SIE ANFANGE
N? J"
430 ACCEPT AT(22,22)VALIDATE("JN")SIZE(-
1)BEEP:A$
440 IF A$="N" THEN 820
450 REM
460 GOSUB 1560 :: IF X=1 THEN BG=1
470 DISPLAY AT(15,18):"IHR ZUG:" :: DISP
LAY AT(18,18):"STEIN";P :: DISPLAY AT(20
,18):"AUF FELD ii"
480 DISPLAY AT(22,1):"COMPUTER ZIEHT DAN
N STEIN";D
490 IF X<7 THEN 520
500 CALL SPRITE(#8,96,11,40*AS(P),(-2+5*
BS(P))*8,#7,116+P*4,2,40*AS(P),(-2+5*BS(
P))*8)
510 CALL LOCATE(#P,193,1)
520 CALL KEY(0,A,H)
530 CALL HCHAR(20,29,79):: CALL HCHAR(20
,29,105)
540 IF A<64 OR A>67 THEN 520
550 CALL HCHAR(20,29,A)
560 IF A=64 THEN DISPLAY AT(20,18):"ABBR
ECHEN ?" :: C=10 :: CALL HCHAR(22,1,32,3
2):: GOTO 660
570 A=A-64
580 CALL KEY(0,B,H)
590 CALL HCHAR(20,30,79):: CALL HCHAR(20
,30,105)
600 IF B<49 OR B>51 THEN 580
610 CALL HCHAR(20,30,B):: B=B-48
620 IF F(A,B)=-P THEN DISPLAY AT(24,1):"
STEIN BITTE VERLEGEN!" ELSE 650
630 CALL SOUND(400,800,6,1000,6)
640 FOR I=1 TO 500 :: NEXT I :: GOSUB 30
00 :: GOTO 470
650 IF F(A,B)<>0 THEN DISPLAY AT(24,1):"
STEIN AUF FREIES FELD SETZEN" :: GOTO 63
0
660 DISPLAY AT(24,1):"BEI IRRTUM >0< SON
ST >ENTER<"
670 CALL KEY(0,I,H)
680 CALL LOCATE(#P,193,1)
690 IF C=10 THEN 710
700 CALL LOCATE(#P,40*A,(-2+5*B)*8)
710 IF H=0 THEN 670
720 IF I=48 THEN C=0 :: CALL LOCATE(#P,1
93,1):: GOTO 470
730 IF I<>13 THEN 670
740 CALL DELSPRITE(#7,#8):: IF C=10 THEN
 1340
750 F(AS(P),BS(P)),F2(AS(P),BS(P))=0 ::
F(A,B),F2(A,B)=-P :: AS(P)=A :: BS(P)=B
760 IF X>1 AND D5<0 THEN 780
770 GOSUB 3140
780 AS$=AS$&Z$(A,B):: BS$=BS$&Z$(A,B)
790 CALL LOCATE(#P,40*A,(-2+5*B)*8):: GO
SUB 2880
800 ON C GOTO 820,1240,1080
810 REM
820 GOSUB 1560 :: IF X=1 THEN BG=2
830 DISPLAY AT(15,18):"COMPUTERZUG" :: D
ISPLAY AT(18,18):"STEIN";P :: DISPLAY AT
(20,18):"AUF FELD qq"
840 IF X<7 THEN 870
850 CALL SPRITE(#8,96,7,40*AC(P),(-2+5*B
C(P))*8,#7,116+P*4,2,40*AC(P),(-2+5*BC(P
))*8)
860 CALL LOCATE(#(P+3),193,1)
870 IF X<5 THEN 900
880 F2(AS(D),BS(D))=0 :: F2(AC(P),BC(P))
=0
890 REM
900 GOSUB 1630 :: C=-100
910 FOR I=1 TO 3 :: FOR J=1 TO 3
920 IF F(I,J)<>0 THEN 950
930 IF C>F1(I,J)THEN 950
940 A=I :: B=J :: C=F1(I,J)
950 NEXT J :: NEXT I
960 IF C>6 THEN S=1 ELSE S=0 :: GOSUB 19
50
970 DISPLAY AT(20,27):CHR$(64+A);CHR$(48
+B)
980 DISPLAY AT(24,22)BEEP:">ENTER<"
990 CALL KEY(0,I,H)
1000 CALL LOCATE(#(P+3),193,1,#(P+3),40*
A,(-2+5*B)*8)
1010 IF I<>13 THEN 990
1020 F(AC(P),BC(P))=0 :: F(A,B),F2(A,B)=
P :: AC(P)=A :: BC(P)=B
1030 CALL DELSPRITE(#7,#8):: CALL LOCATE
(#(3+P),40*A,(-2+5*B)*8)
1040 B$=Z$(A,B):: BS$=BS$&B$ :: IF J=9 O
R S=1 THEN B$=CHR$(ASC(B$)+9)
1050 AS$=AS$&B$ :: GOSUB 2880
1060 ON C GOTO 460,1240,1080
1070 REM
1080 GOSUB 3000 :: L1=LEN(AS$)-3
1090 FOR I=L1 TO 1 STEP -2
1100 B$=SEG$(AS$,I,1)
1110 IF ASC(B$)<73 THEN 1130
1120 NEXT I
1130 BS$=SEG$(BS$,1,I):: L1=I :: GOSUB 2
620 :: GOSUB 2670
1140 M(BG,1)=M(BG,1)+1 :: S$(BG,1,M(BG,1
))=BS$
1150 IF N1<>0 THEN GOSUB 2400 :: GOTO 11
70
1160 GOSUB 2830 :: IF M(BG,1)>1 THEN GOS
UB 2490
1170 A=300 :: FOR I=1 TO 4
1180 FOR J=100 TO 200 STEP 20 :: CALL SO
UND(-400,A+2*J,6):: NEXT J
1190 A=A+200 :: NEXT I
1200 IF C<>10 THEN 1220
1210 DISPLAY AT(22,2):"! ! ! UNENTSCHIED
EN ! ! !" :: GOTO 1350
1220 DISPLAY AT(22,1):"* * SIE HABEN GEW
ONNEN! * *" :: GOTO 1320
1230 REM
1240 GOSUB 3000 :: L1=LEN(BS$)-2 :: GOSU
B 2720 :: GOSUB 2780 :: GOSUB 2400
1250 IF A=1 OR Y=1 THEN 1270
1260 M(BG,2)=M(BG,2)+1 :: S$(BG,2,M(BG,2
))=BS$
1270 GOSUB 2570 :: A=900 :: FOR I=1 TO 4
1280 FOR J=200 TO 100 STEP -20 :: CALL S
OUND(-400,A+2*J,6):: NEXT J
1290 A=A-200 :: NEXT I
1300 DISPLAY AT(22,1):"DER COMPUTER HAT
GEWONNEN!"
1310 IF Y=1 THEN DISPLAY AT(23,1):"VERMU
TLICH HABEN SIE EINEN  FEHLER GEMACHT!"
1320 IF C=2 THEN D=3 :: J=7 ELSE D=0 ::
J=11
1330 FOR I=1 TO 20 :: CALL COLOR(#(1+D),
2,#(2+D),2,#(3+D),2,#(1+D),J,#(2+D),J,#(
3+D),J):: NEXT I
1340 CALL HCHAR(22,1,32,96)
1350 DISPLAY AT(23,4):"N)EUES SPIEL     E
)NDE"
1360 CALL KEY(0,I,H)
1370 IF C=10 THEN 1390
1380 CALL COLOR(#(1+D),2,#(2+D),2,#(3+D)
,2,#(1+D),J,#(2+D),J,#(3+D),J)
```

Scan: S. Höltgen


```
1390 IF H=0 THEN 1360
1400 IF I=69 THEN 1440
1410 IF I<>78 THEN 1360 ELSE GOSUB 3000
1420 CALL HCHAR(6,20,32,12):: GOTO 370
1430 REM
1440 CALL DELSPRITE(ALL):: CALL CLEAR
1450 PRINT "WOLLEN SIE DIE VOM COMPUTER
GELERNTEN PARTIEN":"AUFZEICHNEN? J": : :
1460 ACCEPT AT(21,14)VALIDATE("JN")SIZE(
-1)BEEP:A$
1470 IF A$="N" THEN 1540
1480 H=0 :: GOSUB 3070 :: OPEN #1:E$,OUT
PUT,FIXED 80,INTERNAL
1490 IF H<>0 THEN 1480 ELSE PRINT #1:M(1
,1),M(1,2),M(2,1),M(2,2)
1500 A,B=1 :: GOSUB 3030
1510 B=2 :: GOSUB 3030
1520 A=2 :: B=1 :: GOSUB 3030
1530 B=2 :: GOSUB 3030 :: CLOSE #1
1540 STOP
1550 REM
1560 X=X+1 :: NR=NR+.5 :: IF NR=4 THEN N
R=1
1570 IF X>63 THEN C=10 :: GOTO 1080
1580 P=INT(NR):: GOSUB 3000
1590 DISPLAY AT(6,18):"ZUG";X
1600 D=INT(NR+.5):: IF D=4 THEN D=1
1610 RETURN
1620 REM
1630 FOR I=1 TO 3 :: C,C1=0 :: FOR J=1 T
O 3
1640 IF F2(I,J)=0 THEN 1660
1650 IF F2(I,J)>0 THEN C=C+1 ELSE C1=C1+
1
1660 NEXT J
1670 IF C=2 AND C1=0 THEN R(I)=30 :: GOT
O 1690
1680 IF C1=2 AND C=0 THEN R(I)=7 ELSE R(
I)=0
1690 FOR J=1 TO 3 :: F1(I,J)=R(I):: NEXT
 J
1700 NEXT I
1710 FOR I=1 TO 3 :: C,C1=0 :: FOR J=1 T
O 3
1720 IF F2(J,I)=0 THEN 1740
1730 IF F2(J,I)>0 THEN C=C+1 ELSE C1=C1+
1
1740 NEXT J
1750 IF C=2 AND C1=0 THEN R(I+3)=30 :: G
OTO 1770
1760 IF C1=2 AND C=0 THEN R(I+3)=7 ELSE
R(I+3)=0
1770 FOR J=1 TO 3 :: F1(J,I)=F1(J,I)+R(I
+3):: NEXT J
1780 NEXT I
1790 C,C1=0 :: FOR I=1 TO 3
1800 IF F2(I,I)=0 THEN 1820
1810 IF F2(I,I)>0 THEN C=C+1 ELSE C1=C1+
1
1820 NEXT I
1830 IF C=2 AND C1=0 THEN R(7)=30 :: GOT
O 1850
1840 IF C1=2 AND C=0 THEN R(7)=7 ELSE R(
7)=0
1850 FOR I=1 TO 3 :: F1(I,I)=F1(I,I)+R(7
):: NEXT I
1860 C,C1=0 :: FOR I=1 TO 3
1870 IF F2(I,4-I)=0 THEN 1890
1880 IF F2(I,4-I)>0 THEN C=C+1 ELSE C1=C
1+1
1890 NEXT I
1900 IF C=2 AND C1=0 THEN R(8)=30 :: GOT
O 1920
1910 IF C1=2 AND C=0 THEN R(8)=7 ELSE R(
8)=0
1920 FOR I=1 TO 3 :: F1(I,4-I)=F1(I,4-I)
+R(8):: NEXT I
1930 RETURN
1940 REM
1950 L1=LEN(BS$):: IF L1>8 THEN GOSUB 22
60 ELSE 1970
1960 IF J>0 THEN 2120
1970 IF (M(BG,1)=0 AND M(BG,2)=0)OR X=1
OR D5=1 THEN 2180
1980 REM
1990 G=2 :: FOR I=1 TO M(BG,2)
2000 IF BS$=SEG$(S$(BG,2,I),1,L1)THEN H=
I :: GOTO 2080
2010 IF L1>8 THEN GOSUB 2320 ELSE 2030
2020 IF J>0 THEN 2120
2030 NEXT I
2040 G=1 :: FOR I=1 TO M(BG,1)
2050 IF BS$=SEG$(S$(BG,1,I),1,L1)THEN H=
I :: GOTO 2080
2060 NEXT I :: GOTO 2180
2070 REM
2080 B$=S$(BG,G,H):: L2=LEN(B$)
2090 IF G=1 AND L2-1=L1 THEN N=1 ELSE N=
0
2100 J=ASC(SEG$(B$,L1+1,1))-64
2110 IF N=1 THEN J=J+1 :: S$(BG,G,E)=""
2120 D=Z(J):: A=INT((D-1)/3)+1 :: B=D-(A
-1)*3
2130 IF J<10 THEN 2150
2140 L1=LEN(BS$):: N$=SEG$(BS$,1,L1-1)::
 N1=1 :: J=1 :: GOTO 2120
2150 IF F(A,B)<>0 THEN J=J+1 :: GOTO 212
0
2160 RETURN
2170 REM
2180 IF X=1 OR D5=1 THEN 2230
2190 J=1
2200 D=Z(J):: A=INT((D-1)/3)+1 :: B=D-(A
-1)*3
2210 IF F(A,B)<>0 THEN J=J+1 :: GOTO 220
0 ELSE 2160
2220 REM
2230 A=INT(RND*3+1):: B=INT(RND*3+1)
2240 IF F(A,B)<>0 THEN 2230 ELSE GOSUB 3
140 :: RETURN
2250 REM
2260 B$=SEG$(BS$,L1-5,6)
2270 FOR I=L1-7 TO 1 STEP -2
2280 IF B$<>SEG$(BS$,I,6)THEN 2300
2290 J=ASC(SEG$(BS$,I+6,1))-63 :: RETURN
2300 NEXT I :: J=0 :: RETURN
2310 REM
2320 D$=S$(BG,2,I):: L2=LEN(D$):: IF L2<
8 THEN 2370
2330 FOR C=L2-6 TO 1 STEP -2
2340 IF B$<>SEG$(D$,C,6)THEN 2360
2350 J=ASC(SEG$(D$,C+6,1))-63 :: RETURN
2360 NEXT C
2370 J=0
2380 RETURN
2390 REM
2400 IF N1=0 THEN 2470
2410 L1=LEN(N$)
2420 IF ASC(SEG$(AS$,L1,1))>73 THEN L1=L
1-2 :: GOTO 2420
2430 BS$=SEG$(N$,1,L1):: GOSUB 2670
2440 M(BG,1)=M(BG,1)+1
2450 S$(BG,1,M(BG,1))=SEG$(N$,1,L1)
2460 N1=0 :: GOSUB 2830 :: GOSUB 2490
2470 RETURN
2480 REM
2490 FOR I=1 TO M(BG,1)-1 :: FOR J=I+1 T
O M(BG,1)
2500 L1=LEN(S$(BG,1,I)):: L2=LEN(S$(BG,1
,J))
```

```
2510 IF L1>L2 THEN 2550
2520 IF L1<L2 THEN 2540
2530 IF SEG$(S$(BG,1,I),L1,1)>SEG$(S$(BG
,1,J),L1,1)THEN 2550
2540 B$=S$(BG,1,I):: S$(BG,1,I)=S$(BG,1,
J):: S$(BG,1,J)=B$
2550 NEXT J :: NEXT I :: RETURN
2560 REM
2570 FOR I=1 TO M(BG,2)-1 :: FOR J=I+1 T
O M(BG,2)
2580 IF LEN(S$(BG,2,I))<=LEN(S$(BG,2,J))
THEN 2600
2590 B$=S$(BG,2,I):: S$(BG,2,I)=S$(BG,2,
J):: S$(BG,2,J)=B$
2600 NEXT J :: NEXT I :: RETURN
2610 REM
2620 J=0 :: FOR I=1 TO M(BG,2)
2630 IF SEG$(S$(BG,2,I),1,L1)=BS$ THEN 2
650
2640 J=J+1 :: S$(BG,2,J)=S$(BG,2,I)
2650 NEXT I :: M(BG,2)=J :: RETURN
2660 REM
2670 FOR I=1 TO M(BG,1)
2680 IF SEG$(S$(BG,1,I),1,L1)<>BS$ THEN
2700
2690 S$(BG,1,I)=""
2700 NEXT I :: RETURN
2710 REM
2720 Y=0 :: FOR I=1 TO M(BG,1)
2730 IF SEG$(S$(BG,1,I),1,L1)<>SEG$(BS$,
1,L1)THEN 2750
2740 Y=1
2750 NEXT I
2760 RETURN
2770 REM
2780 A=0 :: FOR I=1 TO M(BG,2)
2790 IF SEG$(S$(BG,2,I),1,L1)<>SEG$(BS$,
1,L1)THEN 2810
2800 A=1 :: DISPLAY AT(23,1):"SO HABEN S
IE SCHON GESPIELT!"
2810 NEXT I :: RETURN
2820 REM
2830 J=0 :: FOR I=1 TO M(BG,1)
2840 IF S$(BG,1,I)="" THEN 2860
2850 J=J+1 :: S$(BG,1,J)=S$(BG,1,I)
2860 NEXT I :: M(BG,1)=J :: RETURN
2870 REM
2880 FOR I=1 TO 3 :: C=0 :: FOR J=1 TO 3
:: C=C+F(I,J):: NEXT J
2890 IF ABS(C)=6 THEN 2970
2900 NEXT I
2910 FOR I=1 TO 3 :: C=0 :: FOR J=1 TO 3
:: C=C+F(J,I):: NEXT J
2920 IF ABS(C)=6 THEN 2970
2930 NEXT I
2940 C=F(1,1)+F(2,2)+F(3,3):: IF ABS(C)=
6 THEN 2970
2950 C=F(1,3)+F(2,2)+F(3,1):: IF ABS(C)=
6 THEN 2970
2960 C=1 :: RETURN
2970 IF C=6 THEN C=2 ELSE C=3
2980 RETURN
2990 REM
3000 FOR I=15 TO 20 :: CALL HCHAR(I,20,3
2,13):: NEXT I
3010 CALL HCHAR(21,1,32,128):: RETURN
3020 REM
3030 FOR I=1 TO M(A,B):: PRINT #1:S$(A,B
,I):: NEXT I :: RETURN
3040 REM
3050 FOR I=1 TO M(A,B):: INPUT #1:S$(A,B
,I):: NEXT I :: RETURN
3060 REM
3070 PRINT : :"SPEICHERGERAET: CS1": : :
: ACCEPT AT(22,17)SIZE(-4):E$
3080 IF E$="CS1" THEN 3110
3090 IF NOT E$="DSK1" OR E$="DSK2" THEN
3070 ELSE PRINT "DATEINAME:"
3100 ACCEPT AT(23,12)SIZE(10):A$ :: IF A
$="" THEN 3090 ELSE E$=E$&"."&A$
3110 ON ERROR 3120 :: RETURN
3120 CALL SOUND(300,900,3,920,3):: PRINT
 "DATEI NICHT GEFUNDEN" :: CALL ERR(H,H
):: RETURN NEXT
3130 REM
3140 D=(A-1)*3+B
3150 IF D=5 THEN D5=1 :: Z$(A,B)="E" ::
GOTO 3220 ELSE D5=-1
3160 IF D<3 THEN RESTORE 3230
3170 IF D=3 OR D=6 THEN RESTORE 3240
3180 IF D=9 OR D=8 THEN RESTORE 3250
3190 IF D=7 OR D=4 THEN RESTORE 3260
3200 FOR I=1 TO 3 :: FOR J=1 TO 3 :: REA
D Z$(I,J):: NEXT J :: NEXT I
3210 FOR I=1 TO 9 :: READ Z(I):: NEXT I
3220 RETURN
3230 DATA A,B,C,D,E,F,G,H,I,1,2,3,4,5,6,
7,8,9
3240 DATA G,D,A,H,E,B,I,F,C,3,6,9,2,5,8,
1,4,7
3250 DATA I,H,G,F,E,D,C,B,A,9,8,7,6,5,4,
3,2,1
3260 DATA C,F,I,B,E,H,A,D,G,7,4,1,8,5,2,
9,6,3
```

Vier gewinnt

Ziel jedes Spielers ist es, als erster vier eigene Chips in eine waagrechte, senkrechte oder diagonale Reihe zu bekommen. Es können zwei Spieler gegen Computer oder Computer gegen Computer spielen (durchschnittliche Zugzeit des Computers: knapp 7 Sekunden). Während des Spieles wird die Spielrunde, der Punktestand und die Zahl der gesetzten Chips angezeigt.

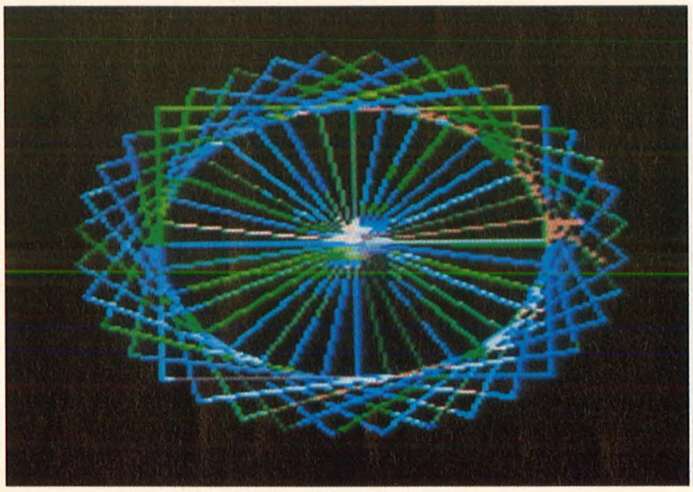
Durch Druck auf die „0"-Taste kann bei jeder Spielzugeingabe oder vor jedem Computerzug wahlweise ein Zug zurückgenommen (also ein Chip entfernt), die Runde abgebrochen und neu angefangen oder ein neues Spiel begonnen werden. Beim Rücksprung zum Ziel ist dann automatisch der Gegenspieler am Zug. Für den Rundensieg gibt es zwei Punkte, für ein „Unentschieden" je einen Punkt für beide Spieler. Bei der Eingabe des Programmes können natürlich alle REMarks ausgelassen werden – aber jetzt viel Spaß, gute Unterhaltung und lassen Sie sich nicht vom Computer unterkriegen!

Günter Schauer

Scan: S. Höltgen

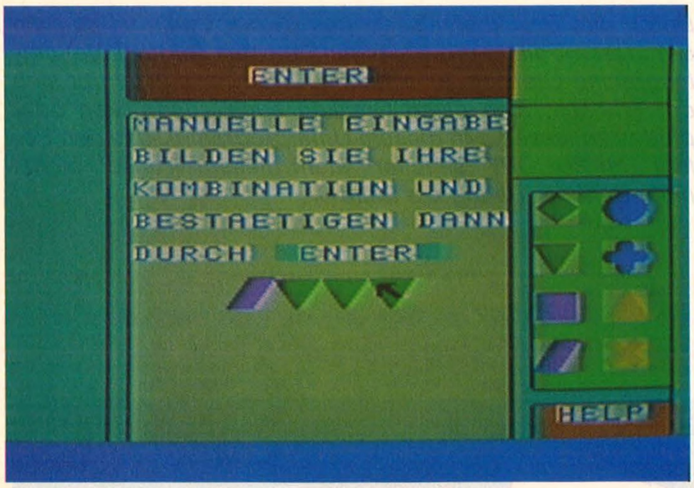
Zum Eintippen . . .

Logo-Interpreter



*Ein Logo-Interpreter in BASIC, mit dem sich spielend leicht tolle Grafiken auf den Bildschirm zaubern lassen (für alle **MSX**-Rechner). Listing auf Seite 69.*

Superhirn



*Ein Denkspiel, das schon in vielen Formen und Varianten existiert, nun auch für den **TI-99/4A**+ Extended-BASIC. Die Programmsteuerung ist wahlweise über Joystick oder Keyboard möglich. Listing auf Seite 70.*

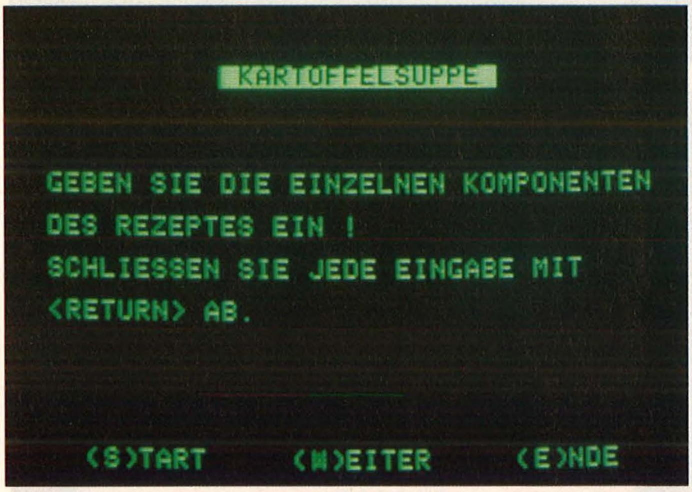
Disk-Sektoren

*Ein Hilfsmittel, das Ihnen Informationen über die Belegung der Diskette durch die darauf abgespeicherten Programme gibt. Zudem wird die Länge von Applesoft- und INTEGER-Programmen, bei Binärprogrammen noch zusätzlich die Startadresse ausgegeben. Die Diskette, die bearbeitet werden soll, muß in Drive 1 liegen (alle **Apple II** mit zwei Diskettenlaufwerken). Listing auf Seite 73.*

Druckerroutinen

*Ein nützliches Hilfsprogramm zur Ansteuerung des NLQ 401 von Schneider. Verschiedene Schriftarten, das Unterstreichen von Text sowie Änderung des Zeilenabstandes stehen zur Auswahl **(Schneider CPC)**. Listing auf Seite 78.*

Menü



*Nun ist es kein Problem mehr, Kochrezepte, die ursprünglich für eine bestimmte Anzahl von Personen ausgelegt waren, nach Bedarf umzurechnen **(Apple II)**. Listing auf Seite 80.*

Scan: S. Höltgen


```
64000 ' ***************************
64010 ' *     LOGO : 1368 Bytes     *
64020 ' *                           *
64030 ' * Sony MSX Computersystem *
64040 ' *           by              *
64050 ' *      Guenter Liesner      *
64090 ' ***************************
64100 FOR Z1=1 TO Z4
64110 IF LO$=B$(Z1) THEN 64140
64120 NEXT
64130 PRINT "Error":END
64140 ON Z1 GOSUB 64160,64220,64240,6
4440,64260,64310,64320,64330,64410,64
470,64390,64420,64430,64480,64340
64150 RETURN
64160 Z6=V1*COS(Z5*PI/180)+X
64170 Z7=V1*SIN(Z5*PI/180)+Y
64180 IF Z2=0 THEN 64200 ELSE IF Z2=2
 THEN V1=Z3:Z3=Z8 ELSE V1=Z3
64190 LINE(X,Y)-(Z6,Z7),Z3:Z3=V1
64200 X=Z6:Y=Z7:IF Z9=0 THEN 64210 EL
SE PUTSPRITE 0,(X,Y),Z3
64210 RETURN
64220 IF Z5+V1>360 THEN Z5=V1-(360-Z5
) ELSE Z5=Z5+V1
64230 RETURN
64240 IF Z5-V1<0 THEN Z5=360-(V1-Z5)
ELSE Z5=Z5-V1
64250 RETURN
64260 V3=V1
64270 Z1=3:V1=180:GOSUB 64140
64280 Z1=1:V1=V3:GOSUB 64140
64290 Z1=3:V1=180:GOSUB 64140
64300 RETURN
64310 Z2=0:RETURN
64320 Z2=1:RETURN
64330 Z3=V1:RETURN
64340 PI=4*ATN(1):X=127:Y=96:Z2=1:Z3=
7:Z9=1:Z5=270:SCREEN 2,1
64350 SPRITE$(0)=CHR$(248)+CHR$(192)+
CHR$(160)+CHR$(144)+CHR$(136)+CHR$(4)
+CHR$(2)+CHR$(1)
64360 LO$="BC":V1=1:GOSUB 64000
64370 OPEN "GRP:" FOR OUTPUT AS #1
64380 RETURN
64390 SCREEN 2:COLOR Z3,V1,V1:CLS:Z8=
V1
64400 RETURN
64410 Z2=2:RETURN
64420 Z9=1:PUTSPRITE 0,(X,Y):RETURN
64430 Z9=0:PUTSPRITE 0,(0,200):RETURN
64440 Z6=V1:Z7=V2:GOSUB 64180
64450 RETURN
64460 PAINT(X,Y),Z3,Z3:RETURN
64470 CIRCLE(X,Y),V1,Z3:RETURN
64480 PRESET(X,Y):COLOR Z3:PRINT #1,V
1$
64490 RETURN
64500 DIM B$(100):RESTORE:Z4=0
64510 Z4=Z4+1:READ B$(Z4):IF B$(Z4)<>
"end" THEN 64510 ELSE RETURN
64520 DATA FD,RT,LT,SP,BK,PU,PD,PC,PE
,CI,BC,ST,HT,PR,TT,end
```

Superhirn

Nach Erscheinen des Titelbildes müssen Eingaben über die Programmführung, die Länge der Kombination, Erstellung der zu suchenden Kombination und der Ausgabe der Anzahl der verschiedenen Symbole in der zu suchenden Kombination eingegeben werden. Dies geschieht, indem man den Pfeil auf die betreffende Stelle fährt und den Feuerknopf, bei Keyboard-Steuerung die Leertaste drückt. Nach diesem Prinzip erfolgen alle Eingaben, begleitet von einem Signalton. Nun erscheint das eigentliche Spielfeld mit den acht verschiedenen Symbolen, dem „ENTER"- und „HELP"-Feld, dem Lösungsfeld sowie drei Anzeigefelder für richtige und richtig positionierte Symbole, den aktuellen Versuch und die Anzahl der verschiedenen Zeichen (ANZVZ) in der zu suchenden Kombination.

Beginnend in der untersten Zeile des Lösungsfeldes werden nun die Symbole plaziert. Ist die Zeile vollständig mit einer Symbolkombination gefüllt, wird „ENTER" betätigt, worauf die Auswertung der Kombination erfolgt. Solange „ENTER" nicht betätigt wurde, können in der Zeile die Symbole überschrieben werden. Ist die Zeile ausgewertet, so wiederholt sich das ganze in der nächsten Zeile.

Insgesamt stehen 20 Versuche zur Verfügung, es werden jedoch immer nur die letzten neun Versuche angezeigt. Durch Betätigen von „HELP" kann das Programm beendet werden, ein Neustart erfolgen, die erstellten Kombinationen in ein Sichtfenster geholt werden oder nachträglich ANZVZ angezeigt werden oder die zu suchenden Kombinationen ausgedruckt werden, wobei jedoch ein Vermerk dieses Ausdruckes erfolgt, falls die richtige Lösung noch nicht gefunden wurde. Soll zum Spiel zurückgekehrt werden, so ist „PLAY" zu betätigen.

Norbert Bleuel

Zeile	Kommentar
100–200	Vorspann
210–260	Aufbau der Variablenliste
260–410	Data-Zeilen
420–590	Charakterdefinitionen
600	Programmstart bei Wiederholung
610–620	Farbzuweisung
630–670	Titelbild
680–830	Eingabe der Startbedingungen
840–880	Spielfeldaufbau 1. Teil
890–950	Festlegung der zu suchenden Zeichenkombination
960–970	Spielfeldaufbau 2. Teil
980	Hauptprogrammaufruf
990–1030	Hauptprogrammsteuerung
1040–1060	Zeichenabgabe
1070–1090	Zeichenaufnahme
1100–1180	Auswertung
1190–1360	Help
1370–1580	Unterprogramme

Programmstruktur

Variable	Bezeichnung
Z$ (n)	Aufnahme der Symbole
GO$ (n)/ GU$ (n)	Symbolkombination
R$ (n)	Ergebnisspeicher (Zusammenfassung von RZ/RP)
G1$/G2$	zu suchende Kombination
RP	Zeichen am richtigen Platz
RZ	richtiges Zeichen
A$	bestimmt ob ANZVZ angezeigt werden soll
K$	Schaltet auf Joystick/Keyboard
A	Länge der Zeichenkombination
L	Spaltenlänge der Zeichenkombination
H	1. Spalte Zeichenkombinationsausdruck
VS	Anzahl der Versuche
V	Zeile Zeichenkombinationsausdruck
AN	ANZVZ
P	Nummer des Symbolsprites
G$, C$, X, J, I, O, T	Variablen zur allgemeinen Verwendung

Variablenliste

```
100 !
110 ! SUPERHIRN
120 !
140 !
150 ! TI 99/4A-Ext.Basic    12 KByte
160 !
170 ! fuer Joystick oder Keyboard
180 !
190 !
200 !
210 GOTO 240 :: CALL SOUND :: CALL CHAR
:: CALL VCHAR :: CALL HCHAR :: CALL GCHA
R :: CALL SPRITE :: CALL MOTION :: CALL
POSITION :: CALL KEY
220 CALL DELSPRITE :: CALL JOYST :: CALL
 HG :: CALL H :: CALL T :: CALL E :: CAL
L M :: CALL SP :: CALL W :: CALL F :: CA
LL J
230 G1$,G2$,A$,K$,G$,C$,LS,RP,RZ,A,AN,VS
,V,H,L,P,X,J,I,O,T
240 OPTION BASE 1 :: RANDOMIZE :: DIM Z$
(16),GO$(20),GU$(20),R$(20)
250 CALL MAGNIFY(3):: CALL CLEAR :: CALL
 COLOR(1,15,15):: CALL SCREEN(6)
260 DATA 7,22,3,5,18,5,4,15,13,2,12,11,1
,8,9,2,4,5,5,2,13,8,1,1
270 !@P-
280 DATA 11,2,3,13,5,15,14,9,1,13,12,7,1
1,15,7,10,18,9
290 DATA 5,5,S,6,7,U,7,9,P,8,11,E,9,13,R
,9,15,H,8,17,I,8,19,R,9,21,N,1,20,BY NOR
,2,26,B
300 DATA 3,27,E,4,28,R,5,28,T,7,28,L,8,2
8,E,9,27,U,10,26,E,11,24,BL,14,21,BCSA08
85
310 DATA PROGRAMMFUEHRUNG@MIT,JOYSTICK\@
PRESS@=BUTTON=,KEYBOARD\@PRESS@=T=
320 DATA BEDIENUNGSHINWEISE,RZ\RICHTIGES
@ZEICHEN,RP\RICHTIGER@PLATZ
330 DATA ANZAHL@DER@ZEICHEN,PRO@ZEILE@^,
@@@@@=4=5=6=7=8=@@@@@
340 DATA SOLL@DER@COMPUTER@EINE,ZEICHENK
OMBINATION,ERSTELLEN@^@=J=N=
350 DATA SOLL@DIE@ANZEIGE@DER,VERSCHIEDE
NEN@ZEICHEN,ERFOLGEN@^@@=J=N=
360 DATA MANUELLE EINGABE,BILDEN SIE IHR
E,KOMBINATION UND,BESTAETIGEN DANN,DURCH
 =ENTER=
370 DATA SICHT,LOESG,NEU,ENDE
380 DATA @SIE@KONNTEN@,**SUPERHIRN**,NIC
HT@LOESEN@
390 DATA ANZVZ,VS\
400 DATA SPIEL@BEENDET,PLAY@GESPERRT
410 DATA KEINE@SICHTUNG,MOEGLICH
420 CALL CHAR(96,"000103070F1F3F7F3F1F0F
07030100000080C0E0F0F8FCFEFFFEFCF8F0E0C080
00")
430 CALL CHAR(100,"00FFFF7F7F3F3F1F1F0F0
F07070303010OFEFEFCFCF8F8F0F0E0E0C0C0808
000")
440 CALL CHAR(104,"007F7F7F7F7F7F7F7F7F7
F7F7F7F7F0000FEFEFEFEFEFEFEFEFEFEFEFEFEF
E00")
450 CALL CHAR(108,"0101030307070F0F1F1F3
F3F7F7FFFFFFFFFFEFEFCFCF8F8F0F0E0E0C0C08
080")
460 CALL CHAR(112,"03071F3F7F7FFFFFFFFFF7
F7F3F1F0F03C0F0F8FCFEFEFFFFFFFFFEFEFCF8F
0C0")
470 CALL CHAR(116,"03070707077FFFFFFFFFF7
F0707070703C0E0E0E0E0FEFFFFFFFFFFEE0E0E0E
0C0")
480 CALL CHAR(120,"000101030307070F0F1F1
F3F3F7F7F0080C0C0E0E0F0F0F8F8FCFCFEFEFFF
F00")
490 CALL CHAR(124,"10387CFE7F3F1F0F1F3F7
FFF7C381000010387CFEFCF8F0E0F0F8FCFE7C381
000")
500 CALL CHAR(140,"FCF0F0F89C8E8703")
510 FOR I=96 TO 127 STEP 4 :: J=J+1 :: Z
$(J)=CHR$(I)&CHR$(I+2):: J=J+1 :: Z$(J)=
CHR$(I+1)&CHR$(I+3):: NEXT I
520 CALL CHAR(40,"0",58,"0",64,"0",128,"
0",136,"0")
530 CALL CHAR(33,"808080808080800FF",46,"
FF80808080808080",137,RPT$("F",16))
540 C$=RPT$("80",8):: CALL CHAR(38,C$,41
,C$,130,C$)
550 C$=RPT$("0",12)&"FF" :: CALL CHAR(35
,"00"&C$,42,"FF"&C$,59,"00"&C$,132,"00"&
C$)
560 C$=RPT$("01",7):: CALL CHAR(36,C$&"0
1",43,C$&"01",37,C$&"FF")
570 C$=RPT$("18",7):: CALL CHAR(61,C$&"1
8",63,C$&"FF")
580 CALL CHAR(133,"00000000FF",44,"81818
18181818181",134,"80808080FF808080")
590 CALL CHAR(92,"0000007C007C",94,"0038
44040810001")
600 CALL DELSPRITE(ALL):: CALL CLEAR ::
G1$,G2$,A$,K$="" :: AN,LS=0 :: FOR I=1 T
O 20 :: GO$(I),GU$(I),R$(I)="" :: NEXT I
610 CALL COLOR(1,15,15,2,2,8,3,2,12,4,2,
12,5,5,16,6,5,16,7,5,16,8,5,16,13,2,4,14
,10,7)
620 CALL COLOR(9,13,15,10,14,15,11,5,15,
12,11,15)
630 RESTORE 260 :: CALL HCHAR(16,1,42,32
):: CALL HG
640 FOR I=1 TO 14 :: CALL SOUND(-500,110
*INT(RND*10+1),0):: READ J,O,X
650 DISPLAY AT(J,O)SIZE(2):Z$(X):: DISPL
AY AT(J+1,O)SIZE(2):Z$(X+1):: NEXT I
660 FOR I=1 TO 10 STEP 2 :: CALL SOUND(-
3000,110,-2+2*I,330,3+I,550,30-2*I):: FO
R J=9 TO 12 :: READ O,X :: CALL H(O,O,1,
X)
670 CALL COLOR(J,15,15):: CALL COLOR(9,1
3,15,10,14,15,11,5,15,12,11,15):: NEXT J
```

Scan: S. Höltgen


```
 :: NEXT I :: CALL H(18,22,2,2)
680 CALL KEY(1,O,T):: IF O=11 THEN K$="T
" ELSE IF O<>18 THEN 680 ELSE CALL SOUND
(50,500,0)
690 IF K$="T" THEN C$="PFEILSTEUERUNG*DU
RCH*DIE*TASTEN*=S=D=E=X=SPACE=*BEI*EINGE
RASTETEM*ALPHALOCK" ELSE C$="=ALPHALOCK*
AUSRASTEN=******"
700 C$=C$&"***WEITER*IM*PROGRAMM*DURCH*D
RUECKEN*EINER*TASTE*ODER*DES*FEUERKNOPFE
S*"
710 CALL HG :: CALL HCHAR(17,5,136,22)::
 CALL H(17,21,2,5):: CALL HCHAR(23,4,42,
26):: L=LEN(C$)
720 I=0 :: J=1 :: X=25
730 G$=SEG$(C$,1+I,J-I):: DISPLAY AT(23,
2+X)SIZE(LEN(G$)):G$ :: CALL SOUND(-200,
-7,15):: X=X-1 :: IF X<0 THEN X=0 :: I=I
+1
740 J=J+1 :: IF J>L THEN J=L
750 CALL KEY(1,O,T):: IF T=0 THEN IF I=L
 THEN 720 ELSE 730
760 CALL SOUND(50,500,0):: CALL HG :: CA
LL H(18,22,2,2):: CALL SPRITE(#1,140,2,1
75,77)
770 CALL J(P,O,T,X,K$):: ON X GOTO 780,7
70,770,770
780 IF (O<169)+(O>176)+(T<69)+(T>148)THE
N 770 ELSE CALL SOUND(50,500,0):: A=INT(
T/16+.74)-1 :: L=A*2 :: H=15-A :: CALL H
G :: CALL H(18,22,2,2)
790 CALL J(P,O,T,X,K$):: ON X GOTO 800,7
90,790,790
800 IF (O<169)+(O>176)+(T<125)+(T>156)TH
EN 790 ELSE CALL SOUND(50,500,0):: X=INT
((T+8)/16+.74):: IF X=9 THEN G$="J"
810 CALL HG :: CALL H(18,22,2,2)
820 CALL J(P,O,T,X,K$):: ON X GOTO 830,8
20,820,820
830 IF (O<169)+(O>176)+(T<125)+(T>156)TH
EN 820 ELSE CALL SOUND(50,500,0):: X=INT
((T+8)/16+.74):: IF X=9 THEN A$="J"
840 CALL CLEAR :: CALL COLOR(4,8,8):: CA
LL VCHAR(1,26,128,144):: CALL VCHAR(1,25
,130,8):: CALL VCHAR(10,25,44,15)
850 CALL VCHAR(1,32,41,24):: CALL HCHAR(
9,25,46):: CALL HCHAR(9,26,42,6):: CALL
HCHAR(4,25,134):: CALL HCHAR(4,26,133,6)
860 CALL HCHAR(22,25,41):: CALL HCHAR(22
,26,42,6):: CALL HCHAR(23,26,137,6):: CA
LL HCHAR(24,26,137,6)
870 CALL VCHAR(1,1,40,24):: CALL VCHAR(1
,2,43,24):: CALL VCHAR(1,3,58,120):: CAL
L VCHAR(1,8,44,24):: CALL HCHAR(4,8,41)
880 CALL HCHAR(4,9,42,16):: CALL HCHAR(4
,3,59,5):: CALL HCHAR(4,5,63):: CALL VCH
AR(5,5,61,20):: CALL T(Z$()):: CALL E ::
 CALL M

890 IF G$="J" THEN 910 ELSE VS=1 :: V=10
 :: CALL H(5,13,2,7):: CALL SP(15,16,H,L
):: CALL COLOR(1,2,15):: GOSUB 1020
900 C$,G1$=GO$(1):: G2$=GU$(1):: GO$(1),
GU$(1)="" :: FOR I=9 TO 24 :: CALL VCHAR
(5,I,32,20):: NEXT I :: GOTO 940
910 FOR I=1 TO A
920 J=INT(RND*15+.5):: IF J/2=INT(J/2)OR
 J<1 THEN 920
930 G1$=G1$&Z$(J):: G2$=G2$&Z$(J+1):: NE
XT I :: C$=G1$
940 FOR I=1 TO L STEP 2 :: G$=SEG$(C$,I,
1):: IF G$="\" THEN 960 ELSE AN=AN+1
950 J=POS(C$,G$,1):: IF J THEN C$=SEG$(C
$,1,J-1)&"\"&SEG$(C$,J+1,L-J):: GOTO 950
960 NEXT I :: CALL SP(5,24,H,L):: CALL C
OLOR(1,2,15,4,2,12)
970 DISPLAY AT(3,1)SIZE(5):"RZ=RP" :: RE
STORE 390 :: CALL H(1,7,6,24):: IF A$="J
" THEN GOSUB 1360
980 V,VS=0 :: GOSUB 990 :: GOTO 1190
990 VS=VS+1 :: IF VS>20 THEN RETURN ELSE
 DISPLAY AT(7,27)SIZE(LEN(STR$(VS))):STR
$(VS):: V=V+2 :: CALL SOUND(50,500,0)
1000 IF V<21 THEN 1020 ELSE V=20 :: CALL
 SP(5,6,H,L)
1010 CALL VCHAR(5,4,58,1):: CALL VCHAR(5
,6,58,1):: FOR I=1 TO 9 :: X=5+2*I :: CA
LL W(GO$(),GU$(),R$(),X,H,L,I,VS):: NEXT
 I
1020 CALL SPRITE(#1,140,2,O,T):: P=0
1030 CALL J(P,O,T,X,K$):: ON X GOTO 1040
,1070,1100,1210
1040 IF (P=0)+(O>208-V*8)+(O<193-V*8)+(T
>128+A*8)+(T<129-A*8)THEN 1030
1050 X=INT((T-1)/8)-1 :: IF (X/2=INT(X/2
)AND A/2=INT(A/2))OR(X/2<>INT(X/2)AND A/
2<>INT(A/2))THEN X=X-1
1060 DISPLAY AT(25-V,X)SIZE(2):Z$(P*2-1)
:: DISPLAY AT(26-V,X)SIZE(2):Z$(P*2)

1070 CALL SOUND(-50,500,0):: IF P THEN C
ALL DELSPRITE(ALL):: P=0 :: GOTO 1020
1080 IF T<224 THEN P=INT((O-73)/24)+1 EL
SE P=INT((O-73)/24)+5
1090 CALL SPRITE(#1+P,92+P*4,1,O,T):: CA
LL COLOR(#2,13,#3,13,#4,14,#5,14,#6,5,#7
,5,#8,11,#9,11):: GOTO 1030
1100 CALL SOUND(50,500,0):: C$="" :: FOR
 I=17-A TO 16+A :: CALL GCHAR(25-V,I,X):
: C$=C$&CHR$(X)
1110 CALL GCHAR(26-V,I,X):: GU$(VS)=GU$(
VS)&CHR$(X):: NEXT I
1120 IF POS(C$,"&",1)THEN GU$(VS)="" ::
CALL SOUND(450,110,0,-7,0):: GOTO 1030 E
LSE GO$(VS)=C$ :: IF G1$="" THEN RETURN
1130 RP,RZ=0 :: FOR I=1 TO L STEP 2 :: G
$=SEG$(G1$,I,1):: J=POS(C$,G$,1)
1140 IF J=0 THEN 1160 ELSE RZ=RZ+1 :: IF
 POS(G1$,G$,J)=J THEN RP=RP+1 ELSE IF PO
S(C$,G$,I)=I THEN J=I :: RP=RP+1
1150 C$=SEG$(C$,1,J-1)&"\"&SEG$(C$,J+1,L
-J)
1160 NEXT I :: R$(VS)=STR$(RZ)&"="&STR$(
RP):: DISPLAY AT(25-V,2)SIZE(3):R$(VS)::
 IF RP<A THEN 990
1170 FOR I=1 TO 4 :: CALL SOUND(50,110,0
):: CALL SOUND(400,220,0,550,6):: NEXT I
 :: CALL SOUND(-2500,330,0,660,0)
1180 CALL SPRITE(#1,140,2,130,220):: DIS
PLAY AT(1,10)SIZE(9):"SUPERHIRN" :: CALL
 HCHAR(2,14,136,5):: DISPLAY AT(3,11)SIZ
E(7):"GELOEST" :: GOTO 1210
1190 CALL E :: RESTORE 380 :: CALL H(1,3
,1,8)
1200 FOR J=2 TO 20 :: FOR I=1240 TO 140
STEP -100 :: CALL SOUND(-50,I,J):: NEXT
I :: NEXT J
1210 CALL COLOR(14,10,10):: DISPLAY AT(2
3,25):"PLAY"
1220 CALL F :: FOR I=12 TO 18 STEP 3 ::
CALL HCHAR(I,26,132,6):: NEXT I :: RESTO
RE 370 :: CALL H(10,19,3,24):: I=VS
1230 CALL J(P,O,T,X,K$):: ON X GOTO 1230
,1240,1280,1340
1240 CALL SOUND(50,550,0):: IF O>144 THE
N CALL CLEAR :: END ELSE IF O>120 THEN 6
00 ELSE IF O<97 THEN J=-1 :: GOTO 1280
1250 CALL E :: DISPLAY AT(1,H)SIZE(L):G1
$ :: DISPLAY AT(2,H)SIZE(L):G2$
1260 DISPLAY AT(3,15)SIZE(7):"LOESUNG" :
```

```
: J=0 :: I=VS :: IF RP=A OR LS THEN 1230
1270 LS=VS :: CALL VCHAR(5,5,137,20):: D
ISPLAY AT(5,24)SIZE(3+LEN(STR$(LS))):"LS
\";STR$(LS):: GOTO 1230
1280 IF (J>-1)+(O<17)+(T<113)THEN 1230 E
LSE IF VS<2 THEN CALL E :: RESTORE 410 :
: CALL H(1,3,2,8):: CALL SOUND(450,220,5
,-7,0):: GOTO 1230
1290 IF J=-1 THEN CALL E :: C$=CHR$(128)
:: DISPLAY AT(3,7)SIZE(16):"VS\";C$;C$;"
,";C$;"<";C$;",";C$;">";C$;",";"AZ" :: J
=-2
1300 CALL SOUND(50,550,0):: IF T>176 AND
 O<25 THEN GOSUB 1360
1310 IF T<137 THEN I=I+1 ELSE IF T>144 A
ND T<169 THEN I=I-1
1320 IF I<1 THEN I=1 ELSE IF I>=VS THEN
I=VS-1
1330 CALL W(GO$(),GU$(),R$(),1,H,L,I,VS)
:: DISPLAY AT(3,10)SIZE(2):STR$(VS-I);C$
 :: GOTO 1230
1340 CALL E :: IF RP=A OR VS>20 THEN RES
TORE 400 :: CALL H(1,3,2,8):: GOTO 1230
1350 J,I=0 :: CALL F :: CALL T(Z$()):: C
ALL M :: CALL COLOR(14,10,7):: GOTO 1030
1360 DISPLAY AT(3,25)SIZE(3):USING "[#]"
:AN :: RETURN
1370 !@P+
1380 SUB E :: FOR I=1 TO 3 :: CALL HCHAR
(I,9,136,16):: NEXT I :: DISPLAY AT(1,2)
SIZE(3):":::" :: SUBEND
1390 SUB M :: DISPLAY AT(2,12)SIZE(5):"E
NTER" :: DISPLAY AT(23,25):"HELP" :: SUB
END
1400 SUB HG :: FOR I=17 TO 24 :: CALL HC
HAR(I,3,128,28):: NEXT I :: SUBEND
1410 SUB H(O,T,I,J):: FOR Z=O TO T STEP
I :: READ C$ :: DISPLAY AT(Z,J)SIZE(LEN(
C$)):C$ :: NEXT Z :: SUBEND
1420 SUB SP(O,T,H,L):: FOR I=O TO T STEP
 2 :: DISPLAY AT(I,H)SIZE(L):RPT$("& ",L
/2-1);"&$"
1430 DISPLAY AT(I+1,H)SIZE(L):RPT$("!#",
L/2-1);"!%" :: NEXT I :: SUBEND
1440 SUB F :: FOR I=26 TO 31 :: CALL VCH
AR(10,I,128,12):: NEXT I :: SUBEND
1450 SUB T(Z$()):: J=10 :: FOR I=1 TO 8
:: DISPLAY AT(J,24)SIZE(2):Z$(I):: DISPL
AY AT(J,27)SIZE(2):Z$(I+8):: J=J+1-(I/2=
INT(I/2)):: NEXT I :: SUBEND
1460 SUB W(GO$(),GU$(),R$(),X,H,L,I,VS)
1470 DISPLAY AT(X,H)SIZE(L):GO$(VS-I)::
DISPLAY AT(X+1,H)SIZE(L):GU$(VS-I):: DIS
PLAY AT(X,2)SIZE(3):R$(VS-I):: SUBEND
1480 SUB J(P,O,T,X,K$)
1490 IF K$="T" THEN 1520
1500 CALL JOYST(1,O,T):: CALL MOTION(#1,
T*-5,O*5,#1+P,T*-5,O*5):: IF O OR T THEN
 1500
1510 CALL KEY(1,O,T):: IF T THEN 1550 EL
SE 1500
1520 CALL KEY(0,O,T):: IF O=83 THEN I=-2
0 ELSE IF O=68 THEN I=20 ELSE IF O=69 TH
EN J=-20 ELSE IF O=88 THEN J=20
1530 IF T=0 THEN I,J=0 ELSE IF (O=32)+(I
=0)+(J=0)=-3 THEN 1550
1540 CALL MOTION(#1,J,I,#1+P,J,I):: GOTO
 1520
1550 CALL POSITION(#1,O,T):: IF (T>248)+
(T<65)+(O>192)THEN 1490
1560 IF T<193 THEN IF O<25 THEN X=3 :: G
OTO 1580 ELSE X=1 :: GOTO 1580
1570 IF T<201 OR O<73 THEN 1490 ELSE IF
O<169 THEN X=2 ELSE IF O>176 THEN X=4 EL
SE 1490
1580 SUBEND
```

Disk-Sektoren

Bedienungsanleitung
Spur Bit Tafel:
Die Spur Bit Tafel informiert den Benutzer über die Belegung der Diskette. Aus Gründen der Übersichtlichkeit wurde hierzu eine grafische Darstellung gewählt. In der Vertikalen sind die Sektoren (0 bis 15), in der Horizontalen die Spuren (0 bis 35) abgetragen. Durch den Stern „*" wird ein freier Sektor, durch den diagonalen Strich „/" ein belegter Sektor angegeben. Somit läßt sich schnell feststellen, wo auf der Diskette noch freier Speicherplatz vorhanden ist.

IDS von einer Datei:
Nach der Eingabe von einem Programmnamen gibt dieser Programmteil die Programmlänge in Byte aus. Bei Binär-Programmen wird außerdem die Startadresse angegeben. Das ist besonders bei der Analyse von Maschinensprache-Programmen von Vorteil (Dieser Programmteil läßt sich natürlich nicht für Textfiles anwenden!).

Spur/Sektor-Liste einer Datei:
Dieser Programmteil gibt die von einem Programm belegten Sektoren aus. Die Ausgabe erfolgt auch hier in grafischer Form. Vertikal sind die Sektoren, horizontal die Spuren abgetragen. Ein belegter Sektor wird durch einen Stern „*" markiert.

Dos Version, Volume Nummer:
Hierüber erhält man folgende allgemeine Informationen von DOS: Die DOS-Version, die Volume-Nummer, die Zahl der Spuren je Diskette, die Zahl der Sektoren je Diskette, die Zahl der Byte je Sektor und schließlich die Maximalzahl der Spur/Sektor-Paare in jedem Sektor der Spur/Sektor-Tabelle.

Programmbeschreibung:
Das Programm macht regen Gebrauch von der RWTS-Routine, die auf der Seite 85 im DOS-Handbuch näher beschrieben ist. Die Daten für die RWTS-Routine befinden sich in den Programmzeilen 2220–2300. Unter Zuhilfenahme dieser Routine werden einzelne Sektoren eingelesen, in den Programmzeilen 250–320 oder der gesamte Katalog (Programmzeilen 680–800). Wie erhält man jetzt aber die Spur Bit Tafel? Zuerst wird durch das Unterprogramm in der Programmzeile 120 ein Rasterbild zur grafischen Ausgabe erstellt. Dann wird die VTOC Volume Table Of Contents), die sich auf der Spur 17 und Sektor 0 befindet, eingelesen. Diese VTOC enthält jeweils 4-Byte-Gruppen für eine Spur. Diese 4 Byte enthalten dann die sogenannte Track Bit Map der jeweiligen Spur. Diese gibt an, welche Sektoren der Spur frei und welche belegt sind. Im Unterprogramm Zahl → Grafik wird die Track Bit Map in ein aussagekräftiges, grafisches Bild umgewandelt.

IDS-Routine:
Hier wird zuerst der gesamte Katalog eingelesen. Dann wird nach dem eingegebenen Programmnamen im Katalog

Biete an Hardware

Markendisketten St. 2,30 DM, CP/M-Banker VG1 75 DM u.a.m. Tel. (0 61 21) 35 43 21, (0 61 45) 3 32 57. G

★★★★★★★★★★★★★★★★
★ **Chip-Shop** ★
★ Spektrum-Hardware ★
★ Commodore-Hardware ★
★ Computer, Monitoren, Drucker ★
★ + Zubehör diverser ★
★ Hersteller. ★
★ Kostenlose Info anfordern bei: ★
★ Chip-Shop, Waldstr. 5, ★
★ 1000 Berlin 21. G ★
★★★★★★★★★★★★★★★★

Tausche oder verkaufe Apple IIe, 2 Lw. + Festplatte + Prehgomader + RAM-Disk + Druckinterface + DOS + Software. Suche 16 bit/IBM-kompatibel, VB 6500 DM. Tel. (02 03) 77 03 00.

DAI-PC, 48K, Doppel-Floppy, Brother EP22, Cass.-Rec., 2 Paddle, DOS u. CP/M, div. Spiele, Textv., Dateiv., Wordstar, VB 2000 DM. Tel. (0 30) 6 64 14 37.

Apple IIc, 5 Mon. alt, Orig.-Monitor, Lit. + Prog.-Pr. 2700 DM. Tel. (0 68 97) 45 16.

QL + QL14 + 44 Cart. m. Softw. + Anleit. + 5 Bücher (Q-DOS, 68008, Basic) + Zubehör, NP 2700 DM, VB 1700 DM. Tel. 0 56 91) 73 23.

Schweiz. Apple II + 100% Comp. 2. Lw., Clock, CPU, Mouse, Speech Card, sep. Tastatur, Monitor, 200 Programme, VB Fr. 2400. Tel. (0 61) 33 22 64.

★★★★★★★★★★★★★★★★
PC-1500/A
Centronics-Interface, Grafik-Spiel, Disassembler, Sprachausgabe, Fast Save/Load.
Tandy M100/200 — Olivetti M10
Pascal, Point/Paint, Hexmonitor, 8—64K-RAM, Assembler, I/O-Port, Hardware-Manual, ROM-Listing, Disassembler, Fast Save/Load, Vier gewinnt, Datei-Monitor, Telefon-Computer, **PGM auf Wunsch.** 0231 Jenz 800325 Wolfgang 801975
★★★★★★★★★★★★★★★★

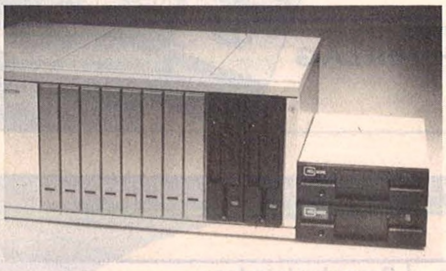
ACHTUNG! TI-99/4A Besitzer......

Compact Peripherie System 99

CPS 99 mit einem Laufwerk 1.698,-
DS DD = 360 K mit 32 K-RAM,
2 x RS 232, Centronics Interface
Disk-Controller DS DD

CPS 99 mit zwei Laufwerken 2.198,-
DS DD = 720 K mit 32 K-RAM
2 x RS 232, Centronics Interface
Disk-Controller DS DD

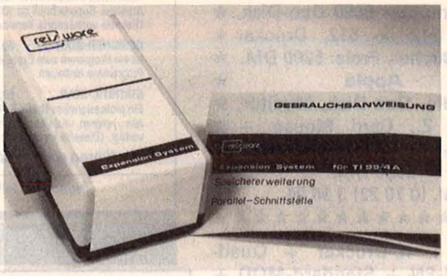
Karten für orig. TI-Box



Alle Erweiterungskarten aus deutscher Fertigung.

32 K-Byte RAM Erweiterung	378,-
RS-232 Karte (parallel/seriell)	358,-
Disk-Steuerkarte DS DD, 4 LW mit Disk-Manager auf Diskette	488,-
Disk-Steuerkarte DS DD, 4 LW mit 32 K-Byte RAM	648,-
RS-232-Karte (parallel/seriell) mit 32 K-Byte RAM	528,-

Externe Erweiterungen



Alle Erweiterungen mit durchgeführtem Datenbus in hochwertiger C-MOS Technik mit umfangreicher deutscher Beschreibung.

32 K RAM Erweiterung	279,-
32 K RAM m. Centronic-Interface	369,-
Centronic-Interface	248,-
RS-232 Schnittstelle	368,-
Sprach-Synthesizer	198,-

NEU - NEU - NEU - NEU - NEU - NEU

Disk-Controller DS DD im Gehäuse mit Netzteil mit Disk-Manager auf Diskette	498,-
Disk-Laufwerk DS DD im Gehäuse mit Netzteil	649,-

Software

Mini Memory	259,-
Editor Assembler	165,-
Multiplan	259,-
TI-Writer	259,-
Terminal Emulator II	85,-
Basic-Compiler	98,-
ID-Data	119,-
ID-Konto	119,-
Spiele	von 30,- bis 79,-

NEU - EXKLUSIV - NEU - EXKLUSIV

GPL-Assembler/Disassembler (Diskette)	149,-
und dazu das Buch von Heiner Martin TI-99/4A intern	38,-

Alle Preise incl. MwSt. zuzügl. 5,- DM Versandkosten. Lieferung per Nachnahme oder Vorkasse.
Ab 200,- DM versandkostenfrei.
Fordern Sie kostenlos unsere Sonderpreisliste an.

Programm-Service
REIS GmbH
D-5584 Bullay
Bergstraße 80
Telefon 06542/2715

Olivetti M24, DVS-Datev-Erweiterung, 256 KB, Typenraddr. 320B, komplett, originalverp., VB 9300 DM. Tel. (0 71 41) 7 13 30.

IBM-Grafikdrucker II, neu 1700 DM. Tel. (0 91 31) 20 77 78.

VG II + Doppel-Floppy, NEWDOS, GDOS, TRS-DOS, CP/M + Programme 1500 DM. Dammers, Tel. (02 21) 70 40 66, 9—18 h.

BROTHER-Typenrad + -Matrix-drucker **HR10 898 DM, M1009sp 669 DM, HR15XL 1388 DM, M1509 1798 DM, HR35 3198 DM, M2024L 3498 DM.** Ulrich Quadt, Heubergredder 14, 2000 Hamburg 60, Tel. (0 40) 7 32 79 46, ab 18 h. G

Basis 108, 128 KB, 2 FD, Monitor, CP/M 3.0, Drucker-Interface, VB 3900 DM. Tel. (0 22 24) 7 12 57, nach 18 h.

★★★★★★★★★★★★★★★★
★ **Frohe Computer-Weih-** ★
★ **nachten!** ★
★ können Sie sich mit meiner ★
★ kompletten Anlage machen. ★
★ Lassen Sie sich überraschen ★
★ (Info: 80 Pf in Briefm.)! Preis: ★
★ VHS. M. Hallupp, Stockumer ★
★ Str. 405, 4600 Dortmund 50, ★
★ Tel. (02 31) 75 64 31, ab 18 h. ★
★★★★★★★★★★★★★★★★

Verkaufe MSX-SVI728 + SVI707 mit Profi-Software, ca. 4 Mon. alt, NP: 2400 DM, VB: 1600 DM. Tel. (0 72 22) 4 74 70.

2 Geräte Altos 8000-2, 11/80, 1/81; 2 Geräte Beehive, 10 DM, 12/80. Tel. (0 51 21) 5 74 22.

EDV-Zubehör-Versand Birg. Wild, Elmshorner Str. 14, 8500 Nürnberg, Tel. (09 11) 34 42 76. G

* * * **Neu Neu Neu** * * *
Btx-Modul!
Neues Medium Home-Btx! **Jetzt einsteigen** — langen Wartezeiten entgehen — **jetzt bestellen,** für C64, Sinclair u. Schneider, nur 258 DM; **Sonderangebot: Disketten** 3" f. CPC 664 u. 6128, 2er Pack nur 25 DM; 5¹/4", ab 3 DM/Stck., Info-Katalog geg. 3 DM in Briefm. M. Meister EDV-Marketing, Rheinstr. 47, 7500 Karlsruhe 21, Tel. (07 21) 55 46 01 (Staack). G

■■■■■■■■■■■■■■■■
■ **Computerkauf** ■
■ **leicht gemacht** ■
■ Wir finanzieren Ihren Compu- ■
■ ter und Zubehör. Info anfor- ■
■ dern! Auch für Händler inter- ■
■ essant. ■
■ **SKG Bank, Postfach 321,** ■
■ **Cecilienstr. 4, 6600 Saar-** ■
■ **brücken, Tel. (06 81)** ■
■ **3 03 01 11.** G ■
■■■■■■■■■■■■■■■■

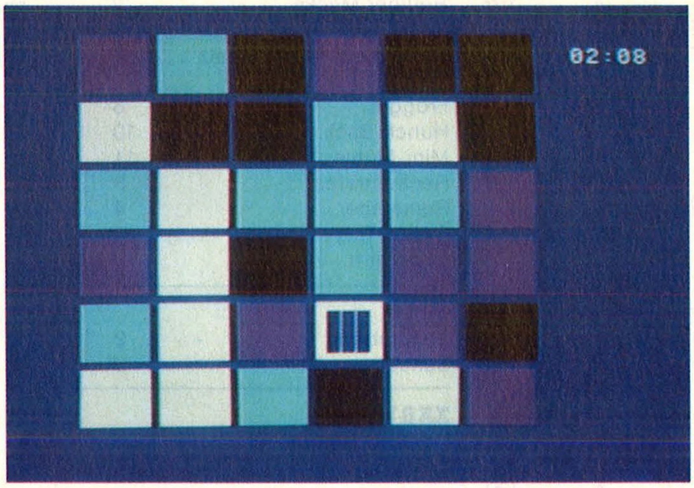
Biete an Software

Videothek (Apple). Die komfort. Verwaltung Ihrer Filme (für VHS, V2000, Beta), 180 DM. Tel. (05 11) 1 71 32.

Scan: S. Höltgen

Zum Eintippen

Magisches Feld

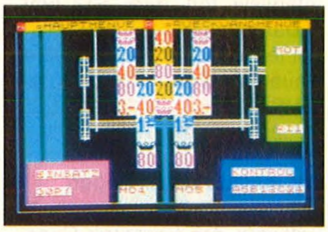


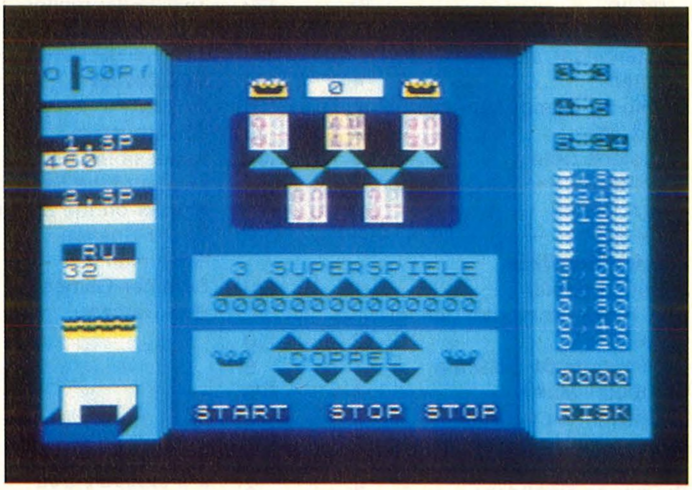
Konzentration und ein gutes Gedächtnis sind nötig, um ein vom **Commodore 64** vorgegebenes Muster aus Quadraten zu rekonstruieren (mit Joysticksteuerung). Listing auf Seite 43.

Zeicheneditor

Ein Vorentwurf auf Papier und ein mühsames Berechnen des Matrixcodes entfällt mit diesem komfortablen Zeicheneditor für den **Schneider CPC464.** Es können gleichzeitig sechs verschiedene Zeichen per Joystick oder Cursorkreuz entworfen werden. Durch viele Befehle, die zum Beispiel Linien und Rechtecke setzen, Bereiche drehen und verschieben, wird die Arbeit extrem einfach. Listing auf Seite 45.

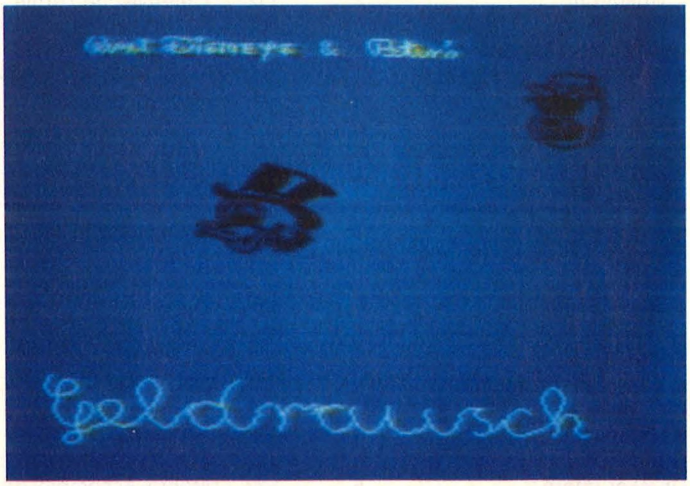
Spielautomat





Der Automaten-Designer für den **Spectrum** (48K) simuliert fast jeden Glücksspiel-Automaten, der in der BRD seit 1960 entstanden ist, oder stellt ihn in seinen Grundzügen dar. Sonderspiele, Risikoleiste und viele weitere Optionen sind selbstverständlich. Der Abruf von fünf festeinprogrammierten Automaten ist jederzeit möglich. Listing auf Seite 54.

Geldrausch



Ein witziges Geschicklichkeitsspiel für den **TI-99/4A** + Extended-BASIC, bei dem Goldtaler so schnell wie möglich einzusammeln sind – doch Vorsicht vor den Panzerknakkern ist geboten. Listing auf Seite 63.

```
7620 GO TO 7600
8000 RETURN
8010 RESTORE 8012: FOR a=USR "A"
 TO USR "U"-1: READ b: POKE a,b:
 NEXT a: FOR a=0 TO 7: READ b: P
OKE USR "U"+a,b: NEXT a: RESTORE
 : RETURN
8012 DATA 0,24,60,102,102,102,10
2,102
8014 DATA 102,102,102,102,102,60
,24,0
8016 DATA 0,8,24,24,56,56,120,24
8018 DATA 24,24,24,24,24,24,60,0
8020 DATA 0,24,60,70,102,102,4,1
2
8021 DATA 8,24,16,48,50,126,70,0
8022 DATA 0,24,60,70,102,102,4,1
2
8023 DATA 12,6,102,102,70,60,24,
0
8024 DATA 0,4,12,12,28,28,60,44
8026 DATA 108,108,126,12,12,12,3
0,0
8028 DATA 0,24,60,102,102,102,10
2,60
8030 DATA 28,50,50,50,109,125,10
9,109
8032 DATA 242,133,229,53,53,181,
226,0
8034 DATA 126,126,0,0,0,192,192,
0
8036 DATA 97,146,146,121,63,63,3
1,0
8038 DATA 134,73,73,158,252,252,
248,0
8040 DATA 1,3,7,15,31,63,127,255
8042 DATA 128,192,224,240,248,25
2,254,255
8044 DATA 0,0,0,0,219,255,126,0
8046 DATA 219,255,126,0,219,255,
126,0
8048 DATA 109,109,50,50,50,28,0,
0
```

Geldrausch

Hatten Sie nicht schon einmal den Wunsch, der reichste Mensch der Welt zu sein? Nun, alles kann dieses Programm zwar nicht, aber Sie können doch zumindest für einige Minuten die reichste Ente der Welt werden, nämlich Dagobert Duck. Wie? Geben Sie zuerst dieses Programm ein, drücken die magischen vier Tasten („R", „U", „N" und die Taste mit dem gelben Punkt) und nehmen den Joystick 1 in die Hand. Wollen Sie Ihr Dagobert-Sein mit einer zweiten Person teilen, so vererben Sie ihrem Mitspieler den anderen Joystick.

Jetzt erscheint dann bald das Titelbild, auf dem Sie sich und Ihre Gegner in voller Größe bewundern können. Dazu erklingt eine Melodie, die Ihre Habgier nur noch steigern kann. Danach wird das Spielfeld aufgebaut, das aus zehn horizontal verlaufenden Bahnen besteht. In jeder Bahn erscheint sogleich entweder ein Taler oder ein Panzerknacker. Und wie könnte bei Dagobert Duck die Devise anders lauten, als „Taler sollst du suchen, Knacker sollst du meiden"? Lassen Sie also Ihrer Raffgier freien Lauf und verschlingen Sie alles, was nicht einem Panzerknacker gleicht. Aber beeilen Sie sich! Sie haben nur einen begrenzten Zeitvorrat zur Verfügung, um alle Wertgegenstände zu vertilgen. Hüten Sie sich außerdem vor den Panzerknackern, denn auch wenn sie sehr freundlich wirken, so sind Begegnungen mit ihnen doch recht schmerzvoll und der Anzahl der Leben äu-

Variable	Bedeutung
Rei ()	Art der Sprites der Bahnen
Pa ()	Zeichen der Sprites der Bahnen
Co ()	Farbe der Sprites der Bahnen
P ()	Punkte
Pun	gewonnene Punkte
Try ()	Anzahl der Versuche
M ()	Dagoberts
tot	Anzahl der ausgeschiedenen Spieler
Ru ()	Level
Spi	Nummer des Spielers
Anz	Anzahl der Spieler
Re	Bahn des Spielers
Ges	Geschwindigkeit der Sprites
Tr	Anzahl der getroffenen Gegenstände
Trm	Anzahl der zu treffenden Gegenstände −1
G	1, wenn im letzten Feld nur Panzerknacker waren
Cou	Zähler für Limit
Limit	Zeitlimit

Variablenliste

Zeilen	Kommentar
310	Dimensionierung
330– 710	Titelbild
730–1940	Hauptprogramm
760– 820	Zeichendatas
830– 840	Zeichendefinitionen
850–1010	Initialisierung, Farben, Bildschirmaufbau
1020–1050	Joystickroutine
1060	Sprite getroffen
1070–1100	Gegenstand getroffen
1110–1160	Panzerknacker getroffen
1170–1190	Game over
1200–1880	Unterprogramme der einzelnen Felder
1890–1940	Alle Felder geschafft
1950–2020	Musikdatas
2030–2050	Sub-Musik
2060–2100	Sub-Display

Für die umfangreiche Grafik des Titelbildes wurden fast alle Zeichen des TI 99/4A umdefiniert. Sie sind in den DATAs des Titelbildes gespeichert. Will man auf das Titelbild verzichten, können die Zeilen 310–710 und die Zeilen 1950–2050 weggelassen werden.

Programmstruktur

ßerst abträglich. Sie haben insgesamt drei Versuche, um das Feld, in dem Sie sich befinden, zu schaffen. Haben Sie alle Versuche verbraucht oder sind Sie fünfmal einem Panzerknacker zu nahe getreten, ist das Spiel für Sie zu Ende. Ansonsten geht es im nächsten Level weiter. Bei jedem neuen Feld ändert sich die Anordnung der Panzerknacker und der Wertgegenstände, bei jedem vierten zusätzlich der Gegenstand. Nach 28 überstandenen Feldern erhalten Sie 100 000 Punkte und einen Extra-Dagobert. Danach beginnt das Spiel wieder bei Level 1.

Das Spiel wird über Joystick gesteuert. Die Anzahl der Versuche wird am unteren Bildrand durch Fähnchen angezeigt. Die Anzeigen für Spieler eins stehen links, die für Spieler zwei rechts. Der Rekord liegt bei etwa 320 000.

Peter Hlawna

```
100 !    **************
110 !    *            *
120 !    * GELDRAUSCH *
130 !    *   VER 1.2  *
140 !    *            *
150 !    **************
160 !    *            *
220 !    *            *
230 !    **************
240 !    *            *
250 !    *  (C) 1985  *
260 !    *     BY     *
270 !    *PETER'S SOFT*
280 !    *            *
290 !    **************
300 CALL CLEAR :: CALL SCREEN(5):: FOR I
=0 TO 14 :: CALL COLOR(I,1,1):: NEXT I
310 DIM REI(11),PA(11),CO(11),HIN(2),P(2
),TRY(2),RU(2),M(2)
320 ! TITELBILD ; SCHRIFT
330 DATA 24424299A5995A24,000000324A7A86
85,00A070202020A010
340 DATA 788710107892711E,10289043545396
13,000000B42A2AA622
350 DATA 000000F242714070,000040A7A8C68D
86,010101,0020505020544834,
360 DATA 0408080810111109,00007088840404
08,0018244242818181,0001020408081010
370 DATA 0907010101020202,30C00000010202
02,0000000384482828,000000C020101010
380 DATA 8142424222424181,00000000000010
1,0000000E71C04080,10202020A0601010,0000
000001020404,000000C011120A0C
390 DATA 000000C324190102,00000000102020
20,0000000202020202,000000F810202010,000
000E01008
400 DATA 814242422424151A,0000000000E010
10,02020101,020204048485661C,28244442820
10E3,1020204080816618,1030484884020 1
410 DATA 0204080808102 0C0,80804040402011
0E,1010102828448201,0C04040404080808,060
60909091020C0
420 DATA 00000000000814 23C,0C020101010122
1C,1818282844840201,0000000001061 8E0,0A1
C244484040808
430 DATA 0808081010100906,04071E22424244
38,408
440 ! PETER'S
450 DATA 7E8191919E515027,080C0F3C8C4D9C
63,0000000089569464,6020400010284850
460 ! DAGOBERT
470 DATA 010103070F0F1F1F,80E0FCFFFCF8F8
F0,0000008060180406,0040C0C0E1FA7F1F
480 DATA 3F3F7FFF0700FEFF,F0E0C08081FB07
F1,1F3F7FFFFFFEF8E0,0080C0E0F0300000
490 DATA 1026294951515357,077088080404C4
E4,FC3E070301010000,C08080E0F0F8FC7C
500 DATA 00000D12120D182E,5737AB6F708030
0C,E8E8D0100C030608,0000101828D021C0,3E7
E7E3C9820F060
510 DATA 19,80711E0808040300,31E26244440
830C0,18D1BE937D0A3200,F0B060A04
520 ! PANZERKNACKER
530 DATA 000000000030C10,000000000F00F
00,0000000000008040,0000060908040201
540 DATA 101024A36018079C,000031C20C7082
06,4040806090501060
550 DATA 0F1409010101000 1,F7FF9C9CDEDEF7
73,8CD8F0F0E0C0C080,9010080808080810
560 DATA 0204040408090804,0077F8F87003FD
88,008040106 8A61890,1010102020202040
570 DATA 04040301,447FFF00E01807,F0E0800
10E7080,404080
580 RESTORE :: FOR I=33 TO 86 :: READ A$
 :: CALL CHAR(I,A$):: NEXT I
590 FOR I=96 TO 136 :: READ A$ :: CALL C
HAR(I,A$):: NEXT I
600 ! BILDSCHIRMAUFBAU
610 FOR I=96 TO 136 :: READ A,B :: CALL
HCHAR(A,B,I):: NEXT I
620 DATA 8,14,8,15,8,16,9,13,9,14,9,15,9
,16,9,17
630 DATA 10,13,10,14,10,15,10,16,11,12,1
1,13,11,14,11,15,11,16,12,12,12,13,12,14
,12,15,12,16
640 DATA 2,26,2,27,2,28,3,25,3,26,3,27,3
,28,4,25,4,26,4,27,4,28
650 DATA 5,25,5,26,5,27,5,28,6,25,6,26,6
,27,6,28
660 DISPLAY AT(1,6):"!"&CHR$(34)&"#$%&'(
 * STUV"
670 CALL HCHAR(2,14,41)
680 DISPLAY AT(20,4):",-  .  /
  ." :: DISPLAY AT(21,4):"0123456789:6;<
 ;8=8>?@"
690 DISPLAY AT(22,4):"ABCDEFGHFI GHJKHFL
MNOP" :: DISPLAY AT(23,5):"QR"
700 FOR I=1 TO 8 :: CALL COLOR(I,11,1,MI
N(I+8,14),2,1):: NEXT I
710 RESTORE 1950 :: FOR I=1 TO 29 :: CAL
L MUSIK :: NEXT I
720 ! HAUPTPROGRAMM
730 CALL SCREEN(2):: CALL CLEAR :: CALL
CHARSET :: CALL MAGNIFY(3)
740 ! ZEICHENDATAS
750 ! DAGOBERT
760 DATA 0C1E3F7FFC3B275F,5F1F3F1F3F1F0F
,000020C080C04050,54FCF8E000E08
770 DATA 000004030103020A,2A3F1F07000701
,3078FCFE3FDCE4FA,FAF8FCF8FCF8F0
780 DATA ,000E1F3F1F3FFF,,000000C0A0A0FC
790 ! PANZERKNACKER
800 DATA 00001F3104313F3F,3F1F1F0C0703,0
000E0104010F8F8,9898F810F0E0
810 ! BAHN
820 DATA 00000000000000FF
830 RESTORE 760 :: FOR I=96 TO 112 :: RE
AD A$ :: CALL CHAR(I,A$):: NEXT I
840 CALL CHAR(124,"68F0E85CB47F3C10",128
,"E0407F7C70404040")
850 RANDOMIZE :: TOT,HIN(1),HIN(2),P(1),
P(2),PUN,TRY(1),TRY(2)=0 :: RU(1),RU(2),
SPI=1 :: M(1),M(2)=5
860 CALL COLOR(11,12,1,12,4,1,13,6,1)::
FOR I=1 TO 8 :: CALL COLOR(I,16,1):: NEX
T I
870 FOR I=2 TO 22 STEP 2 :: CALL HCHAR(I
,1,112,32):: NEXT I
```

```
880 DISPLAY AT(11,7):"1 ODER 2 SPIELER ?
"
890 CALL KEY(0,K,S):: IF K<49 OR K>50 TH
EN 890 ELSE ANZ=K-48
900 CALL HCHAR(11,1,32,32)
910 RE=11 :: CALL SPRITE(#1,96,9,160,128
)
920 DISPLAY AT(24,1)SIZE(5):USING "|:##"
:M(1):: DISPLAY AT(24,24)SIZE(5):USING "
|:##":M(2)
930 CALL DISPLAY(1,P(1)):: CALL DISPLAY(
2,P(2))
940 IF TR>TRM OR G=1 THEN 950 ELSE IF TR
Y(SPI)=3 THEN 1170 ELSE TRY(SPI)=TRY(SPI
)+1 :: GOTO 960
950 RU(SPI)=RU(SPI)+1 :: G=0 :: TRY(SPI)
=1 :: P(SPI)=P(SPI)+RU(SPI)*(LIMIT-COU)+
500*G :: CALL DISPLAY(SPI,P(SPI))
960 CALL HCHAR(24,10,128,TRY(1)):: CALL
HCHAR(24,20,128,TRY(2)):: CALL HCHAR(24,
10+TRY(1),32,2):: CALL HCHAR(24,20+TRY(2
),32,2)
970 FOR I=2 TO 12 :: CALL DELSPRITE(#I):
: NEXT I :: CALL MOTION(#1,0,0):: COU=0
980 TR=0
990 IF RU(SPI)>16 THEN 1000 ELSE ON RU(S
PI)GOSUB 1200,1240,1260,1280,1300,1340,1
360,1380,1400,1440,1460,1280,1480,1520,1
540,1560 :: GOTO 1010
1000 ON RU(SPI)-16 GOSUB 1580,1620,1640,
1660,1690,1730,1750,1770,1790,1830,1850,
1870,1890
1010 GES=16 :: FOR I=2 TO 11 :: CALL SPR
ITE(#I,PA(I),CO(I),I*16-16,256,0,GES)::
GES=-GES :: NEXT I
1020 CALL JOYST(SPI,XJ,YJ):: CALL MOTION
(#1,0,7*XJ)
1030 IF YJ THEN RE=MAX(MIN(RE-YJ/4,11),2
):: CALL POSITION(#1,Y,X):: CALL LOCATE(
#1,16*(RE-1),X)
1040 IF XJ THEN CALL PATTERN(#1,98-XJ/2)
1050 COU=COU+1 :: IF COU=LIMIT THEN 940
ELSE CALL COINC(#1,#RE,15,A):: IF A=0 TH
EN 1020
1060 IF REI(RE)=3 THEN 1020 ELSE IF REI(
RE)=2 THEN 1110
1070 CALL MOTION(#RE,0,0):: CALL COLOR(#
RE,4):: CALL PATTERN(#RE,120):: CALL SOU
ND(-100,12000,6,300,6,-2,6)
1080 P(SPI)=P(SPI)+PUN :: TR=TR+1 :: CAL
L DISPLAY(SPI,P(SPI))
1090 IF TR>TRM THEN 940
1100 REI(RE)=3 :: GOTO 1020
1110 CALL MOTION(#1,0,0):: CALL PATTERN(
#1,104):: CALL SOUND(300,110,6,127,6,-8,
0)
1120 FOR I=1 TO 5 :: CALL SOUND(100,6000
,6,200,6,1100,6,-1,0):: CALL SOUND(100,1
10,6,127,6,440,6,-4,6):: NEXT I
1130 CALL DELSPRITE(ALL):: M(SPI)=M(SPI)
-1 :: IF M(SPI)=0 OR TRY(SPI)=3 THEN 117
0
1140 IF HIN(2+(SPI=2))=0 THEN SPI=SPI+1
:: IF SPI>ANZ THEN SPI=1
1150 DISPLAY AT(11,7):"ACHTUNG SPIELER";
SPI :: FOR I=1 TO 700 :: NEXT I :: CALL
HCHAR(11,1,32,32)
1160 G=0 :: GOTO 910
1170 CALL DELSPRITE(ALL):: DISPLAY AT(11
,5):"GAME OVER SPIELER";SPI
1180 FOR I=1 TO 1000 :: NEXT I :: CALL H
CHAR(11,1,32,32):: HIN(SPI)=1 :: TOT=TOT
+1 :: M(SPI)=0 :: IF TOT<ANZ THEN 1140
1190 CALL DELSPRITE(ALL):: GOTO 850
1200 CALL CHAR(116,"0000030F1F1F3C3E3E1E
1F0F030000000000080E0F0F078F8F8F0F0E0C")
1210 CALL CHAR(120,"0000FFFF000206020202
00FFFF0000000000FFFF0070505050507000FFFF")
1220 FOR I=2 TO 10 STEP 2 :: REI(I)=1 ::
 REI(I+1)=2 :: PA(I)=116 :: PA(I+1)=108
:: CO(I)=12 :: CO(I+1)=7 :: NEXT I
1230 PUN=10 :: TRM=4 :: LIMIT=60 :: RETU
RN
1240 GOSUB 1200 :: FOR I=2 TO 6 :: REI(I
)=1 :: PA(I)=116 :: CO(I)=12 :: REI(I+5)
=2 :: PA(I+5)=108 :: CO(I+5)=7 :: NEXT I
1250 RETURN
1260 GOSUB 1200 :: FOR I=2 TO 11 :: REI(
I)=1 :: PA(I)=116 :: CO(I)=12 :: NEXT I
:: FOR I=5 TO 9 :: REI(I)=2 :: PA(I)=108
 :: CO(I)=7 :: NEXT I
1270 RETURN
1280 GOSUB 1200 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I
1290 TRM=0 :: LIMIT=150 :: G=1 :: RETURN
1300 CALL CHAR(116,"000000FFC1DFDCDEDEDE
DFC1FF0000000000000FE06FA7AFAFAFAFA06FE")
1310 CALL CHAR(120,"0000FFFF000E080E020E
00FFFF0000000000FFFF0070505050507000FFFF")
1320 FOR I=2 TO 10 STEP 2 :: REI(I)=1 ::
 REI(I+1)=2 :: PA(I)=116 :: PA(I+1)=108
:: CO(I)=3 :: CO(I+1)=7 :: NEXT I
1330 PUN=50 :: TRM=4 :: LIMIT=60 :: RETU
RN
1340 GOSUB 1300 :: FOR I=2 TO 11 :: REI(
I)=1 :: PA(I)=116 :: CO(I)=3 :: NEXT I
1350 LIMIT=80 :: TRM=9 :: RETURN
1360 GOSUB 1300 :: FOR I=2 TO 6 :: REI(I
)=1 :: PA(I)=116 :: CO(I)=3 :: REI(I+5)=
2 :: PA(I+5)=108 :: CO(I+5)=7 :: NEXT I
1370 LIMIT=55 :: TRM=4 :: RETURN
1380 GOSUB 1300 :: FOR I=2 TO 11 :: REI(
I)=1 :: PA(I)=116 :: CO(I)=3 :: NEXT I :
: FOR I=5 TO 9 :: REI(I)=2 :: PA(I)=108
:: CO(I)=7 :: NEXT I
1390 RETURN
1400 CALL CHAR(116,"070301030F1F3C3A7A7C
7E7A3C1F0F03C0800080E0F078BCFC7CBCBC78F0
E080")
1410 CALL CHAR(120,"0000FFFF000B1A0A0A0B
00FFFF0000000000FFFF00B8A8A8A8B800FFFF")
1420 FOR I=2 TO 11 :: REI(I)=2 :: PA(I)=
108 :: CO(I)=7 :: NEXT I :: FOR I=5 TO 9
 STEP 2 :: REI(I)=1 :: PA(I)=116 :: CO(I
)=12 :: NEXT I
1430 TRM=2 :: LIMIT=35 :: PUN=100 :: RET
URN
1440 GOSUB 1400 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: FOR I=3 TO 9 STEP 3 :: REI(I)=1 :: PA(
I)=116 :: CO(I)=12 :: NEXT I
1450 LIMIT=35 :: RETURN
1460 GOSUB 1400 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: FOR I=4 TO 9 :: REI(I)=1 :: PA(I)=116
:: CO(I)=12 :: NEXT I
1470 REI(6),REI(7)=2 :: PA(6),PA(7)=108
:: CO(6),CO(7)=7 :: TRM=3 :: LIMIT=43 ::
 RETURN
1480 CALL CHAR(116,"000000000003070F1020
4080FF0000000000000000F0E8DC3E1F0E04F8")
1490 CALL CHAR(120,"0000FFFF000B1A0A0A0B
00FFFF0000000000FFFF00B8A8A8A8B800FFFF")
1500 FOR I=2 TO 11 STEP 2 :: REI(I)=2 ::
 REI(I+1)=1 :: PA(I)=108 :: PA(I+1)=116
:: CO(I)=7 :: CO(I+1)=11 :: NEXT I
1510 TRM=4 :: LIMIT=48 :: PUN=100 :: RET
URN
1520 GOSUB 1480 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: FOR I=5 TO 8 :: REI(I)=1 :: PA(I)=116
:: CO(I)=11 :: NEXT I
```

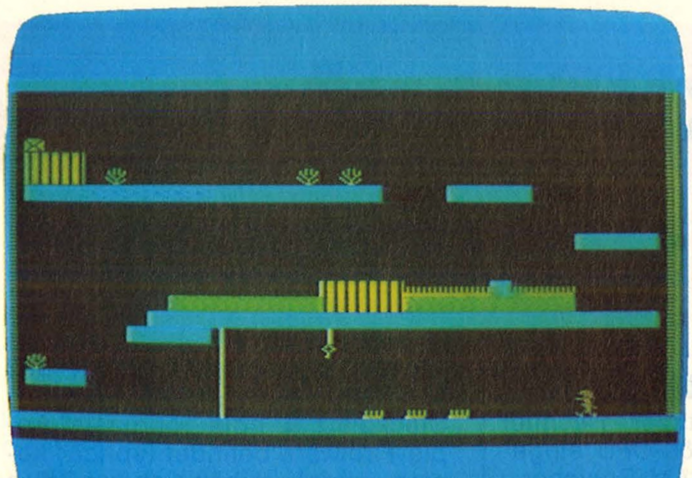
Scan: S. Höltgen


```
1530 TRM=3 :: LIMIT=40 :: RETURN
1540 GOSUB 1480 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: FOR I=3 TO 9 STEP 3 :: REI(I)=1 :: PA(
I)=116 :: CO(I)=11 :: NEXT I
1550 TRM=2 :: LIMIT=33 :: RETURN
1560 GOSUB 1480 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: REI(3),REI(6),REI(7),REI(10)=1 :: PA(3
),PA(6),PA(7),PA(10)=116
1570 CO(3),CO(6),CO(7),CO(10)=11 :: TRM=
3 :: LIMIT=40 :: RETURN
1580 CALL CHAR(116,"0001030A1224443F1F0F
07030100000000C0F028241211FEFCF8F0E0C08"
)
1590 CALL CHAR(120,"0000FFFF000B1A0A0A0B
00FFFF0000000000FFFF00B8A8A8A8B800FFFF")
1600 FOR I=3 TO 10 :: REI(I)=2 :: PA(I)=
108 :: CO(I)=7 :: NEXT I :: FOR I=2 TO 1
1 STEP 3 :: REI(I)=1 :: PA(I)=116 :: CO(
I)=INT(RND*14)+3
1610 NEXT I :: LIMIT=38 :: TRM=3 :: PUN=
100 :: RETURN
1620 GOSUB 1580 :: FOR I=2 TO 10 :: REI(
I)=1 :: PA(I)=116 :: CO(I)=INT(RND*14)+3
 :: NEXT I :: FOR I=2 TO 11 STEP 3 :: RE
I(I)=2 :: PA(I)=108 :: CO(I)=7
1630 NEXT I :: TRM=5 :: LIMIT=53 :: RETU
RN
1640 GOSUB 1580 :: FOR I=2 TO 11 :: REI(
I)=1 :: PA(I)=116 :: CO(I)=INT(RND*14)+3
 :: NEXT I :: FOR I=4 TO 9 :: REI(I)=2 :
: PA(I)=108 :: CO(I)=7 :: NEXT I
1650 TRM=3 :: LIMIT=38 :: RETURN
1660 GOSUB 1580 :: FOR I=3 TO 10 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I
1670 REI(2),REI(6),REI(7),REI(11)=1 :: P
A(2),PA(6),PA(7),PA(11)=116 :: CO(2),CO(
6),CO(7),CO(11)=INT(RND*14)+3
1680 RETURN
1690 CALL CHAR(116,"001F1F1F001F11151511
1F001F1F1F0000F8F8F800F858585848F800F8F8
F8")
1700 CALL CHAR(120,"0000FFFF001D111D051D
00FFFF0000000000FFFF00DC545454DC00FFFF")
1710 FOR I=2 TO 11 :: REI(I)=2 :: PA(I)=
108 :: CO(I)=7 :: NEXT I :: FOR I=4 TO 9
 :: REI(I)=1 :: PA(I)=116 :: CO(I)=14 ::
 NEXT I
1720 REI(6),REI(7)=2 :: PA(6),PA(7)=108
:: CO(6),CO(7)=7 :: TRM=3 :: PUN=500 ::
LIMIT=35 :: RETURN
1730 GOSUB 1690 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: REI(3),REI(6),REI(7),REI(10)=1 :: PA(3
),PA(6),PA(7),PA(10)=116
1740 CO(3),CO(6),CO(7),CO(10)=14 :: RETU
RN
1750 GOSUB 1690 :: FOR I=3 TO 10 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: FOR I=2 TO 11 STEP 3 :: REI(I)=1 :: PA
(I)=116 :: CO(I)=14 :: NEXT I
1760 RETURN
1770 GOSUB 1690 :: FOR I=2 TO 11 :: REI(
I)=2 :: PA(I)=108 :: CO(I)=7 :: NEXT I :
: REI(3),REI(5),REI(8),REI(10)=1 :: PA(3
),PA(5),PA(8),PA(10)=116
1780 CO(3),CO(5),CO(8),CO(10)=14 :: RETU
RN
1790 CALL CHAR(116,"001F151F171E151C151E
171F151F000000FC54FCF43C541C543CF4FC54FC
")
1800 CALL CHAR(120,"0000FFFF002E6A2A2A2E
00FFFF0000000000FFFF00EEAAAAAAEE00FFFF")
1810 FOR I=2 TO 10 STEP 2 :: REI(I)=2 ::
 REI(I+1)=1 :: PA(I)=108 :: PA(I+1)=116
:: CO(I)=7 :: CO(I+1)=15 :: NEXT I
1820 PUN=1000 :: LIMIT=42 :: TRM=4 :: RE
TURN
1830 GOSUB 1790 :: FOR I=2 TO 11 :: REI(
I)=1 :: PA(I)=116 :: CO(I)=15 :: NEXT I
:: FOR I=4 TO 9 :: REI(I)=2 :: PA(I)=108
 :: CO(I)=7 :: NEXT I
1840 LIMIT=32 :: TRM=3 :: RETURN
1850 GOSUB 1790 :: FOR I=2 TO 11 :: REI(
I)=1 :: CO(I)=15 :: NEXT I
:: REI(3),REI(5),REI(8),REI(10)=2 :: PA(
3),PA(5),PA(8),PA(10)=108
1860 CO(3),CO(5),CO(8),CO(10)=7 :: TRM=5
 :: LIMIT=45 :: RETURN
1870 GOSUB 1790 :: FOR I=2 TO 11 :: REI(
I)=1 :: PA(I)=116 :: CO(I)=15 :: NEXT I
1880 LIMIT=67 :: TRM=9 :: RETURN
1890 FOR I=110 TO 660 STEP 110 :: FOR J=
0 TO 20 STEP 2 :: CALL SOUND(-200,I+110,
J):: CALL SOUND(-200,I,J):: NEXT J :: NE
XT I
1900 Q=1 :: FOR I=1 TO 30 :: Q=-Q :: IF
Q=-1 THEN CALL PATTERN(#1,96)ELSE CALL P
ATTERN(#1,100)
1910 NEXT I
1920 DISPLAY AT(9,11):"100.000" :: DISPL
AY AT(13,1):"FUER VOLLENDEN ALLER FELDER
"
1930 P(SPI)=P(SPI)+100000 :: CALL DISPLA
Y(SPI,P(SPI))
1940 FOR I=1 TO 1000 :: NEXT I :: RU(SPI
)=1 :: M(SPI)=M(SPI)+1 :: CALL HCHAR(9,1
,32,32):: CALL HCHAR(13,1,32,32):: GOTO
910
1950 DATA 6,392,247,6,392,294,6,494,196,
6,494,294
1960 DATA 6,440,262,6,440,294,6,392,247,
6,392,294
1970 DATA 6,494,196,6,494,294,6,587,247,
6,587,294
1980 DATA 6,523,220,6,523,294,6,494,196,
6,494,294
1990 DATA 6,587,247,6,587,294,13,659,262
2000 DATA 6,523,220,6,523,294,13,587,247
2010 DATA 6,494,196,6,494,294,6,523,220,
6,523,294
2020 DATA 6,440,262,6,440,294,20,392,196
2030 SUB MUSIK
2040 READ A,B,C :: FOR I=0 TO A STEP 1 :
: CALL SOUND(-200,B,I,C,I+9):: NEXT I
2050 SUBEND
2060 SUB DISPLAY(SPI,P)
2070 IF SPI=2 THEN 2090
2080 DISPLAY AT(1,1)SIZE(7):STR$(P):: SU
BEXIT
2090 DISPLAY AT(1,22):USING "#######":P
2100 SUBEND
```

Scan: S. Höltgen

Zum Eintippen

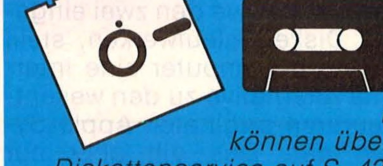
Miner



*Ein kleiner sprunggewaltiger Bergarbeiter sammelt am Abend Nachrichten auf, die sein Chef hinterlegt hat. Um an die Briefe heranzukommen, muß er einige Klippen überwinden (**TI-99/4A** in Grundversion). Listing auf Seite 43.*

Work-Tool 64

*Mit dieser Erweiterung für den **C64** können Sie fehlende Befehle im Commodore-BASIC (MERGE, OLD, Fehlerkanal der Floppy . . .), die sich sonst nur durch lange POKE-Schlangen ausführen ließen, anwenden. Listing auf Seite 46.*

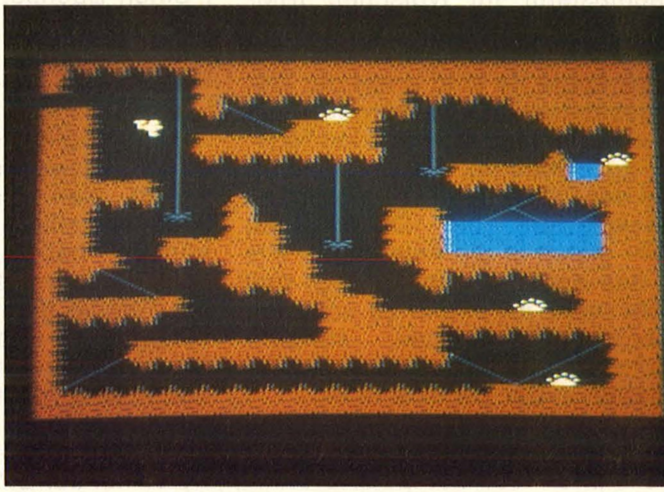


Listings mit diesen Symbolen können über den Kassetten- und Diskettenservice auf S. 42 angefordert werden.

Analog-Uhr

*Der **Schneider CPC** ist als Analoguhr einsetzbar und zeigt zusätzlich die vollen Stunden durch ein akustisches Signal an. Listing auf Seite 48.*

Caveman



Mit Hilfe eines kleinen Hubschraubers versucht der Held dieses Spiels, die wertvollen Schätze, die in den Höhlen verborgen sind, einzusammeln (Atari 600 XL, 800 XL und 130 XE). Listing auf Seite 51.

Miner

Zunächst stehen Sam Pflanzen, Wände und andere Hindernisse im Weg. Es kommen dann Fließbänder und Sümpfe hinzu. Nur mit Schlüssel und Hammer lassen sich die Türen zu den gesuchten Briefen öffnen. Zu allem Überfluß hat das Programm vor den Erfolg noch ein Skelett gesetzt.

Nachdem das Programm mit RUN gestartet worden ist, erscheint das Titelbild. Nach Drücken der Taste „s" kann der Spieler zwischen zwei Schwierigkeitsstufen wählen, danach beginnt das Spiel. Die Steuerung übernehmen folgende Tasten:

Sprung links W
Sprung E
Sprung rechts R
Links S
Rechts D

Nach drei Fehlern, die unten links als Grabsteine angezeigt werden, ist das Spiel beendet. Es erscheint das Schlußbild und die erreichten Punkte werden angezeigt.
Wenn es gelingt, den vierten und letzten Brief zu bekommen, ist das Spiel ebenfalls vorbei. Der Punktestand wird genannt und das Titelbild erscheint wieder.
Das Programm kann nur mit CLEAR unterbrochen werden. *Dirk Schübel*

```
100 CALL CLEAR
110 SW=0
120 PU=0
130 GR=0
140 CALL CHAR(33,"7F7F60606060607F")
150 CALL CHAR(127,"FFFFFFFFFFFFFFFF")
160 CALL CHAR(34,"FCFC0C0C0C0000F8")
170 CALL CHAR(35,"7F00006060607F7F")
180 CALL CHAR(36,"F818181818181F8F8")
190 CALL CHAR(37,"7F7F60606060607F")
200 CALL CHAR(38,"F8F818181818F8")
210 CALL CHAR(39,"7F60606060606060")
220 CALL CHAR(40,"F818181818181818")
230 CALL CHAR(41,"607070786C676360")
240 CALL CHAR(42,"18383878D8981818")
250 CALL CHAR(43,"6060606060606060")
260 CALL CHAR(44,"1818181818181818")
270 FOR I=1 TO 12
280 CALL COLOR(I,11,2)
290 NEXT I
300 CALL SCREEN(5)
310 S=110
320 FOR I=1 TO 12
330 S=S+50
340 CALL SOUND(-400,110,5,S,0)
350 READ A,B,C
360 CALL HCHAR(A,B,C)
370 DATA 12,13,33,12,14,34,13,13,35,13,1
4,36,12,15,37,12,16,38,13,15,39,13,16,40
380 DATA 12,17,41,12,18,42,13,17,43,13,1
8,44
390 NEXT I
400 CALL HCHAR(10,8,127,16)
410 CALL HCHAR(15,8,127,16)
420 CALL VCHAR(10,7,127,6)
430 CALL VCHAR(10,24,127,6)
440 PRINT " W=SPR  LINKS E=SPRUNG
R=SPR  RECHTS S=LINKS        D=RECHTS    "
450 PRINT " START MIT <S>"
460 Y=22
470 CALL KEY(1,K,S)
480 IF K=2 THEN 500
490 GOTO 470
500 CALL CLEAR
510 PRINT "SCHWIERIGKEITSGRADE";::::;" 1.
 LEICHT";:::;" 2. SCHWER";:::::::;"WAEHLEN
 SIE";:::::
520 CALL KEY(1,K,S)
530 IF S=0 THEN 520
540 IF K=19 THEN 570
550 IF K=7 THEN 560 ELSE 520
560 SW=1
570 X=30
580 CALL CLEAR
590 CALL CHAR(128,"003C7E1E16FC1C0C")
600 CALL CHAR(95,"10284428101030")
610 CALL CHAR(129,"003C7E78683F3830")
620 CALL CHAR(94,"1C1E0F1F3260C080")
630 CALL CHAR(88,"1455552294492A1C")
640 CALL CHAR(90,"007EBDDBE7DBBD00")
650 CALL CHAR(130,"3C1E0D0C1CA56321")
660 CALL CHAR(131,"3C78B03038A5C684")
670 CALL CHAR(132,"003C7E3C5A3C2418")
680 CALL CHAR(133,"7E5A5A5A3C242466")
690 CALL CHAR(136,"00FFFFFFFFFFFFFF")
700 CALL CHAR(137,"00FFFF66E3FF7EFF")
710 CALL CHAR(134,"FF7E7C38280808")
720 CALL CHAR(58,"0078ECC6EEEEFEFE")
730 CALL CHAR(59,"F8C0F8C0F8C0F8C0")
740 CALL CHAR(60,"1F031F031F031F03")
750 CALL CHAR(135,"42E7425A7E5A7E66")
760 CALL CHAR(138,"7E7E7E7E7E7E7E7E")
770 CALL CHAR(143,"00FFFF66E3FF7EFF")
780 CALL CHAR(144,"00007E7E7E7E")
790 CALL CHAR(159,"5FEDFFBBFFDDBB")
800 CALL CHAR(91,"DDDDDDDDDDDDDDDD")
810 CALL CHAR(93,"1818181818181818")
820 CALL CHAR(33,"C080583C6C3C1812")
830 CALL CHAR(34,"141E141218A5C264")
840 CALL CHAR(35,"03011A3C363C1848")
850 CALL CHAR(36,"2878284818A54346")
860 CALL CHAR(92,"000000555555FFFF")
870 CALL COLOR(13,9,2)
880 CALL COLOR(8,11,2)
890 CALL COLOR(14,8,5)
900 CALL COLOR(16,4,13)
910 CALL CLEAR
920 REM  HOEHLE 1
930 HO=1
940 MA=3
950 FOR I=1 TO 26
960 READ A,B,C,D
970 CALL HCHAR(A,B,C,D)
980 DATA 23,1,136,32,16,8,136,24,17,16,9
3,1,18,16,95,1,15,9,159,7,15,16,91,4,14,
16,91,4,15,20,159,8
990 DATA 14,20,92,8,20,1,136,4,17,7,136,
5,14,24,136,2,11,28,136,4,8,1,136,18
1000 DATA 8,22,136,4,1,1,136,32,7,17,88,
1,7,15,88,1,7,2,91,3,6,2,91,3,5,2,90,1
1010 DATA 22,18,92,1,22,20,92,1,22,22,92
,1,8,5,137,6,7,6,88,1
1020 NEXT I
1030 CALL VCHAR(2,1,59,21)
1040 CALL VCHAR(2,32,60,21)
1050 CALL VCHAR(17,11,93,6)
1060 Y=22
1070 X=28
1080 CALL HCHAR(24,2,58,GR)
1090 IF SW=0 THEN 1840
1100 CALL HCHAR(19,2,88)
1110 CALL HCHAR(14,25,92)
1120 GOTO 1840
1130 REM  HOEHLE 2
1140 CALL CLEAR
1150 FOR I=1 TO 28
```

Scan: S. Höltgen


```
1160 READ A,B,C,D
1170 CALL HCHAR(A,B,C,D)
1180 DATA 23,1,136,32,20,1,136,9,17,1,13
6,6,16,13,136,19,15,15,159,14,15,29,91,3
1190 DATA 10,7,136,6,9,8,136,6,14,1,136,
3,14,29,91,3,17,12,136,6,11,6,136,6,8,9,
136,6,8,15,137,18
1200 DATA 8,25,32,2,14,23,88,1,1,1,136,3
2,9,17,93,1,10,17,95,1,7,14,88,1,7,17,88
,1,2,15,134,1
1210 CALL HCHAR(24,5,58,GR)
1220 DATA 3,15,95,1,2,13,136,3,3,14,134,
1,7,12,88,1,7,19,88,1,22,3,90,1
1230 NEXT I
1240 HO=2
1250 X=27
1260 Y=22
1270 CALL VCHAR(2,1,59,21)
1280 CALL VCHAR(2,32,60,21)
1290 CALL VCHAR(21,4,93,2)
1300 CALL VCHAR(21,9,93,2)
1310 SCH=2
1320 IF SW=0 THEN 1840
1330 CALL HCHAR(23,21,137,5)
1340 CALL HCHAR(22,20,88)
1350 GOTO 1840
1360 REM  HOEHLE 3
1370 CALL CLEAR
1380 FOR I=1 TO 26
1390 READ A,B,C,D
1400 CALL HCHAR(A,B,C,D)
1410 DATA 23,1,136,32,1,1,136,32,20,1,13
6,6,17,7,136,2,16,8,136,2,15,9,136,2,14,
10,136,2
1420 DATA 13,12,136,2,12,13,136,2,11,14,
136,12,18,7,134,1,16,19,136,14,14,1,136,
4,7,1,136,4
1430 DATA 21,16,33,1,22,16,34,1,13,1,159
,4,8,2,95,1,5,5,136,12,8,18,136,6,8,28,1
36,4,2,23,136,3
1440 DATA 15,20,94,1,8,24,136,5,3,23,95,
1,22,2,90,1
1450 NEXT I
1460 CALL VCHAR(21,6,138,2)
1470 CALL VCHAR(2,1,59,21)
1480 CALL VCHAR(2,32,60,21)
1490 CALL VCHAR(7,5,138,7)
1500 CALL VCHAR(2,21,138,6)
1510 CALL HCHAR(24,2,58,GR)
1520 SCH=2
1530 G=1
1540 J=16
1550 HO=3
1560 X=27
1570 Y=22
1580 IF SW=0 THEN 1840
1590 CALL HCHAR(15,25,88)
1600 GOTO 1840
1610 REM  HOEHLE 4
1620 CALL CLEAR
1630 HO=4
1640 Y=22
1650 X=27
1660 SCH=2
1670 FOR I=1 TO 27
1680 READ A,B,C,D
1690 CALL HCHAR(A,B,C,D)
1700 DATA 1,1,136,32,23,1,136,32,21,3,13
6,3,10,2,136,5,17,25,136,3,17,8,136,4,14
,2,136,4
1710 DATA 15,16,136,2,16,17,136,2,17,19,
136,2,18,20,136,2,19,21,136,2,20,22,136,
2,11,17,136,15
1720 DATA 10,4,136,6,7,2,136,4,20,3,159,
3,22,1,92,7,4,7,137,25,14,12,136,5,11,9,
136,3
1730 DATA 12,26,95,1,22,17,88,1,10,21,88
,1,15,4,95,1,11,27,32,1,3,28,90,1
1740 NEXT I
1750 CALL VCHAR(2,1,59,21)
1760 CALL VCHAR(2,32,60,21)
1770 CALL VCHAR(15,12,93,8)
1780 CALL VCHAR(2,5,93,5)
1790 CALL VCHAR(12,25,93,5)
1800 CALL HCHAR(24,2,58,GR)
1810 IF SW=0 THEN 1840
1820 CALL HCHAR(17,25,137,2)
1830 CALL HCHAR(23,9,137,2)
1840 REM  SAM SETZEN
1850 Z=130
1860 V=128
1870 CALL HCHAR(Y,X,Z)
1880 CALL HCHAR(Y-1,X,V)
1890 REM  BEWEGUNG
1900 CALL KEY(1,K,S)
1910 CALL SOUND(20,-2,1)
1920 IF HO<>3 THEN 1940
1930 GOTO 4340
1940 IF K=2 THEN 2070
1950 IF K=3 THEN 2210
1960 IF K=5 THEN 2350
1970 IF K=4 THEN 2580
1980 IF K=6 THEN 2960
1990 CALL GCHAR(Y+1,X,A)
2000 IF A=159 THEN 2240
2010 IF A=143 THEN 4220
2020 IF A=137 THEN 2050
2030 CALL SOUND(15,-1,0)
2040 GOTO 1900
2050 CALL HCHAR(Y+1,X,32)
2060 GOTO 1900
2070 REM  LINKS
2080 Z=130
2090 V=128
2100 X=X-1
2110 CALL HCHAR(Y-1,X+1,32)
2120 CALL HCHAR(Y,X+1,32)
2130 CALL HCHAR(Y-1,X,V)
2140 CALL GCHAR(Y,X,A)
2150 IF A<>32 THEN 4020
2160 CALL HCHAR(Y,X,Z)
2170 CALL GCHAR(Y+1,X,A)
2180 IF A=137 THEN 4200
2190 IF A<>136 THEN 3510
2200 GOTO 1900
2210 REM  RECHTS
2220 Z=131
2230 V=129
2240 X=X+1
2250 CALL HCHAR(Y-1,X-1,32)
2260 CALL HCHAR(Y,X-1,32)
2270 CALL HCHAR(Y-1,X,V)
2280 CALL GCHAR(Y,X,A)
2290 IF A<>32 THEN 4020
2300 CALL HCHAR(Y,X,Z)
2310 CALL GCHAR(Y+1,X,A)
2320 IF A=137 THEN 4200
2330 IF A<>136 THEN 3510
2340 GOTO 1900
2350 REM  SPRUNG
2360 Z=133
2370 V=132
2380 P=Y
2390 FOR I=1 TO 3
2400 CALL HCHAR(Y-I,X,Z)
2410 CALL HCHAR(P,X,32)
2420 CALL GCHAR(Y-1-I,X,A)
2430 IF A<>32 THEN 3660
2440 CALL HCHAR(Y-1-I,X,V)
2450 P=P-1
2460 NEXT I
2470 Y=Y-4
```

Scan: S. Höltgen


```
2480 P=Y
2490 FOR I=1 TO 4
2500 CALL HCHAR(Y+I-1,X,V)
2510 CALL HCHAR(P-1,X,32)
2520 CALL HCHAR(Y+I,X,Z)
2530 P=P+1
2540 NEXT I
2550 IF HO=0 THEN 4930
2560 Y=Y+4
2570 GOTO 1900
2580 REM  SPRUNG LINKS
2590 I=0
2600 V=128
2610 Z=130
2620 P=Y
2630 U=P
2640 Y=Y-1
2650 IF Y<U-3 THEN 2770
2660 X=X-1
2670 CALL HCHAR(P,X+1,32)
2680 CALL GCHAR(Y,X,A)
2690 IF A<>32 THEN 3660
2700 CALL HCHAR(P-1,X+1,32)
2710 CALL GCHAR(Y-I-1,X,A)
2720 IF A<>32 THEN 3660
2730 CALL HCHAR(Y-I,X,Z)
2740 CALL HCHAR(Y-1-I,X,V)
2750 P=P-1
2760 GOTO 2640
2770 Y=Y+1
2780 X=X-1
2790 CALL GCHAR(Y,X,A)
2800 IF A<>32 THEN 4020
2810 CALL GCHAR(Y+1,X,A)
2820 IF A=137 THEN 4200
2830 IF A<>32 THEN 3310
2840 CALL HCHAR(P-2,X+1,32)
2850 CALL HCHAR(P-1,X+1,32)
2860 CALL HCHAR(P,X+1,32)
2870 CALL GCHAR(Y-1,X,A)
2880 IF A<>32 THEN 3660
2890 CALL HCHAR(Y,X,Z)
2900 CALL HCHAR(Y-1,X,V)
2910 P=P+1
2920 GOTO 2770
2930 Y=U
2940 X=X+1
2950 GOTO 1900
2960 REM  SPRUNG RECHTS
2970 V=129
2980 Z=131
2990 P=Y
3000 U=Y
3010 Y=Y-1
3020 IF Y<U-3 THEN 3140
3030 X=X+1
3040 CALL HCHAR(P,X-1,32)
3050 CALL HCHAR(P-1,X-1,32)
3060 CALL GCHAR(Y-1,X,A)
3070 IF A<>32 THEN 3660
3080 CALL GCHAR(Y,X,A)
3090 IF A<>32 THEN 3660
3100 CALL HCHAR(Y,X,Z)
3110 CALL HCHAR(Y-1,X,V)
3120 P=P-1
3130 GOTO 3010
3140 Y=Y+1
3150 X=X+1
3160 CALL GCHAR(Y,X,A)
3170 IF A<>32 THEN 4020
3180 CALL GCHAR(Y+1,X,A)
3190 IF A<>32 THEN 3430
3200 CALL HCHAR(P-2,X-1,32)
3210 CALL HCHAR(P-1,X-1,32)
3220 CALL HCHAR(P,X-1,32)
3230 CALL HCHAR(Y,X,Z)
3240 CALL HCHAR(Y-1,X,V)
3250 P=P+1
3260 GOTO 3140
3270 Y=U
3280 X=X-1
3290 GOTO 1900
3300 REM  AUFL.
3310 IF A=88 THEN 4020
3320 IF A=94 THEN 4180
3330 IF A=90 THEN 4020
3340 IF A=33 THEN 4020
3350 IF A=35 THEN 4020
3360 CALL HCHAR(P-2,X+1,32)
3370 CALL HCHAR(P-1,X+1,32)
3380 CALL HCHAR(P,X+1,32)
3390 CALL HCHAR(Y,X,Z)
3400 CALL HCHAR(Y-1,X,V)
3410 IF A=92 THEN 4020
3420 GOTO 1900
3430 CALL HCHAR(P-2,X-1,32)
3440 CALL HCHAR(P-1,X-1,32)
3450 CALL HCHAR(P,X-1,32)
3460 CALL HCHAR(Y,X,Z)
3470 CALL HCHAR(Y-1,X,V)
3480 IF A=92 THEN 4020
3490 IF A=88 THEN 4020
3500 GOTO 1900
3510 IF A=92 THEN 4020
3520 IF A<>32 THEN 1900
3530 V=132
3540 Z=133
3550 Y=Y+1
3560 CALL HCHAR(Y-2,X,32)
3570 CALL HCHAR(Y-1,X,32)
3580 CALL HCHAR(Y-1,X,V)
3590 CALL HCHAR(Y,X,Z)
3600 CALL GCHAR(Y+1,X,A)
3610 IF A<>32 THEN 3630
3620 GOTO 3550
3630 REM
3640 IF HO=0 THEN 4930
3650 GOTO 1900
3660 IF A=95 THEN 3720
3670 CALL SOUND(1000,110,0)
3680 RESTORE 980
3690 GR=GR+1
3700 IF GR>3 THEN 4730
3710 GOTO 4610
3720 PU=PU+100
3730 IF HO=2 THEN 3790
3740 IF HO=3 THEN 3870
3750 IF HO=4 THEN 3940
3760 CALL VCHAR(21,11,32,2)
3770 CALL SOUND(400,-8,0)
3780 GOTO 2440
3790 SCH=SCH-1
3800 IF SCH=0 THEN 3840
3810 CALL SOUND(400,-8,0)
3820 CALL VCHAR(21,9,32,2)
3830 GOTO 2440
3840 CALL SOUND(500,-7,0)
3850 CALL VCHAR(21,4,32,2)
3860 GOTO 2440
3870 SCH=SCH-1
3880 CALL SOUND(400,-8,0)
3890 IF SCH=0 THEN 3930
3900 CALL VCHAR(7,5,32,7)
3910 CALL HCHAR(8,24,32,4)
3920 GOTO 2440
3930 CALL VCHAR(21,6,32,2)
3940 CALL SOUND(400,-8,0)
3950 SCH=SCH-1
3960 IF SCH=0 THEN 3990
3970 CALL VCHAR(21,12,32,2)
3980 GOTO 2440
3990 CALL VCHAR(2,4,32,5)
```

```
4000 GOTO 2440
4010 GOTO 2440
4020 IF A=90 THEN 4080
4030 IF A=94 THEN 4180
4040 CALL SOUND(1000,110,0)
4050 GR=GR+1
4060 IF GR>3 THEN 4730
4070 GOTO 4610
4080 CALL SOUND(500,110,0)
4090 CALL SOUND(500,220,0)
4100 CALL SOUND(750,330,0)
4110 PU=PU+1000
4120 Y=22
4130 X=27
4140 IF HO=2 THEN 1370
4150 IF HO=3 THEN 1620
4160 IF HO=4 THEN 5010
4170 GOTO 1140
4180 CALL HCHAR(7,3,32,2)
4190 GOTO 1900
4200 CALL HCHAR(Y+1,X,143)
4210 GOTO 1900
4220 CALL HCHAR(Y-1,X,32)
4230 V=135
4240 CALL HCHAR(Y,X,V)
4250 CALL SOUND(-200,110,0)
4260 CALL SOUND(-500,262,0)
4270 FOR I=1 TO 400
4280 NEXT I
4290 RESTORE 980
4300 GR=GR+1
4310 IF GR>3 THEN 4730
4320 GOTO 4610
4330 END
4340 REM  GEIST
4350 ON G GOTO 4430,4520
4360 CALL HCHAR(21,A,32)
4370 CALL HCHAR(22,A,32)
4380 CALL GCHAR(21,J,B)
4390 IF B<>32 THEN 3690
4400 CALL HCHAR(21,J,Q)
4410 CALL HCHAR(22,J,W)
4420 GOTO 1940
4430 J=J+1
4440 A=J-1
4450 Q=33
4460 W=34
4470 IF J>14 THEN 4500
4480 G=1
4490 GOTO 4360
4500 G=2
4510 GOTO 4360
4520 J=J-1
4530 A=J+1
4540 Q=35
4550 W=36
4560 IF J=10 THEN 4590
4570 G=2
4580 GOTO 4360
4590 G=1
4600 GOTO 4360
4610 IF HO=1 THEN 4650
4620 IF HO=2 THEN 4670
4630 IF HO=3 THEN 4690
4640 IF HO=4 THEN 4710
4650 RESTORE 980
4660 GOTO 910
4670 RESTORE 1180
4680 GOTO 1140
4690 RESTORE 1410
4700 GOTO 1370
4710 RESTORE 1700
4720 GOTO 1620
4730 REM
4740 HO=0
4750 RESTORE 4810
4760 REM  SCHLUSSBILD
4770 CALL CLEAR
4780 FOR I=1 TO 11
4790 READ A,B,C,D
4800 CALL HCHAR(A,B,C,D)
4810 DATA 23,1,136,32,15,6,136,9,14,6,15
9,9,18,15,91,4,19,15,91,4,20,15,91,4,21,
15,91,4,22,15,91,4
4820 DATA 22,5,88,1,22,11,88,1,22,25,88,
1
4830 NEXT I
4840 Y=12
4850 FOR X=7 TO 15
4860 CALL SOUND(50,-6,0)
4870 CALL VCHAR(Y,X-1,32,2)
4880 CALL HCHAR(Y,X,132)
4890 CALL HCHAR(Y+1,X,133)
4900 NEXT X
4910 X=X-1
4920 GOTO 3530
4930 CALL HCHAR(16,15,32)
4940 CALL HCHAR(17,15,132)
4950 CALL HCHAR(17,15,32)
4960 PRINT " SIE HABEN ";PU;"PUNKTE ";::
;" ERREICHT "
4970 FOR I=1 TO 1000
4980 NEXT I
4990 RESTORE 320
5000 GOTO 100
5010 CALL CLEAR
5020 PRINT "HERZLICHEN GLUECKWUNSCH ";::
;"SIE HABEN ES GESCHAFT ";:::;"SIE ERREIC
HTEN ";PU;"PUNKTE";:::
5030 FOR I=1 TO 1000
5040 NEXT I
5050 RESTORE 320
5060 GOTO 100
```

Work-Tool 64

Die BASIC-Version wird mit LOAD,,WORK-TOOL 64 B",8 geladen und mit RUN gestartet. Nach wenigen Sekunden erscheint in der linken oberen Ecke ,,SYS 49152". Durch Drücken der RETURN-Taste wird das Maschinenprogramm gestartet. Die Eingabe von NEW ist hier nicht nötig. Nach jedem Start durch SYS 49152 erscheint die Meldung:

**** WORK-TOOL 64 ****
1985 (C) PETER HEINTZEN

Ihnen stehen dann neun verschiedene neue Befehle zur Verfügung.

@F: Liest den Fehlerkanal der Floppy und zeigt ihn auf dem Bildschirm an.

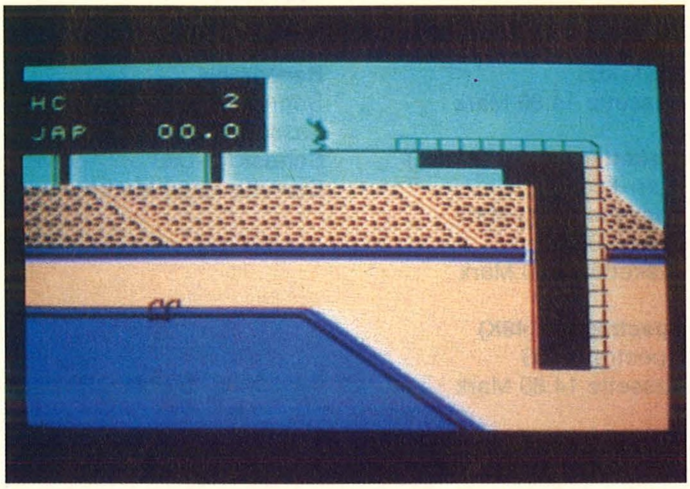
@$: Liest die Directory und zeigt sie auf dem Bildschirm an.

@O: (OLD) Holt durch NEW gelöschte Programme wieder zurück. Nach einem Reset muß erst die Erweiterung neu gestartet werden, bevor diese Routine den alten Zustand wieder herstellen kann.

@R: (RESET OFF) Programme, bei denen durch zum Beispiel 788,52 und POKE 792,193

Scan: S. Höltgen

Turmspringen



*Drei Versuche hat der Athlet auf dem **TI-99/4A** + Extended-BASIC, um mit einem perfekten Sprung vom Brett die höchste Punktzahl zu erreichen. Listing auf Seite 64.*

List-Formatter

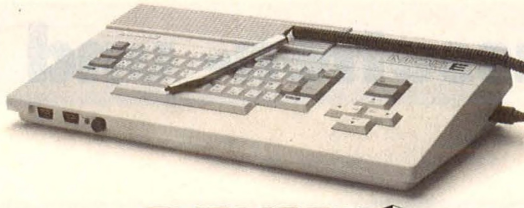
*Dieses Maschinenprogramm für den **Apple II** ermöglicht es dem Programmierer, seine Programmlistings formatiert auszudrucken. Listing auf Seite 67.*

Automatisches Bandarchiv

*Der Kassettenberg für den **Schneider CPC 464** wird immer größer. Die Frage „Was ist auf welcher Kassette?" stellt sich bei jedem Arbeiten mit dem CPC 464 aufs neue. Das automatische Bandarchiv macht nun damit Schluß. Listing auf Seite 68.*

Multicolorhardcopy

*Mit diesem Hardcopyprogramm für den **C64** ist es möglich, ein Multicolorbild in DIN A4 genauso auszudrucken, wie es auf einem Schwarzweißbildschirm aussieht. Die Grautöne für die 16 Farben lassen sich verändern, und das Maschinenprogramm ist frei verschiebbar. Ein zusätzliches kleines BASIC-Programm kann die Bilder der gängigsten Zeichenprogramme (Koalapainter, Blazing Paddles ...) laden und drucken. Listing auf Seite 70.*



THOMSON

Und hier gibt's die Computer, die man mit einem Griff im Griff hat.

1000 Berlin: Hertie (12), Wilmersdorfer Str. 118 · Wertheim (15), Kurfürstendamm 231 · Hertie (20), Carl-Schurz-Str. 24 KaDeWe (30), Tauentzienstr. 21–24 Wertheim (41), Schloßstr. 11–15, Hertie (44), Karl-Marx-Str. 92–98
2000 Hamburg: Horten, Mönckebergstr. 1 Alsterhaus (36), Jungfernstieg 16–20 · Otto Versand,
2300 Kiel: Hertie, Sophienblatt 2
2800 Bremen: Horten (1), Papenstr. 5
2940 Wilhelmshaven: Tiemann, Marktstr. 52
3000 Hannover: Horten, Seilwinderstr. 8 Saturn, Spinnereistr. 16 · Trend Data Comp., Am Marstall 18–22 · Brinkmann (91), Georgstr. 10
3180 Wolfsburg: Computer-Zentrum, Rothenfelderstr. 12
3200 Hildesheim: Horten, Almsstr. 41
3300 Braunschweig: Horten, Bohlweg 72
3320 Salzgitter/Lebenstedt: Fricke Computer, Berliner Str. 54
3500 Kassel: Heini Weber, Wilhelmstr. 1
4000 Düsseldorf: Horten, Berliner Allee 52 · Radio Evertz, Königsallee 63–65 Data Becker, Merowingerstr. 30
4030 Ratingen: Elektro-Funk-Studio, Bechemerstr. 22–32
4100 Duisburg: Horten, Düsseldorfer Str. 32
4150 Krefeld: Horten, Ostwall 170–180
4300 Essen: Horten, Kettwiger Str. 1a · Schossau, Kopstadtplatz 10
4400 Münster: Horten, Ludgeristr. 1
4402 Greven: Radio-Hildebrandt, Marktplatz 1
4500 Osnabrück: Hertie, Neumarkt 13 · Horten, Wittekindstr. 23 · Radio-Rohlfing, Grossestr. 24
4530 Ibbenbüren: Medialand Overmeyer, Unterer Markt 6
4600 Dortmund: Horten, Hansastr. 5
4628 Lünen: Koch-Interfunkmarkt, Langestr. 62
4700 Hamm: Horten, Bahnhofsplatz 3
4800 Bielefeld: Horten (1), Stresemannstr. 11
4837 Verl: Beckhoff, Uhlandstraße 2
5000 Köln: Kaufhof, Hohe Str. 41 · Saturn, Hansaring 97 · VIA Computer, Apostelnstr. 9–11 · Hertie, Neumarkt 1
5090 Leverkusen: Radio Winzen, Kölner Str. 67–69 · City-Computer, Doenhoffstr. 23
5100 Aachen: Allo Pach GmbH, Adalbertstr. 82 · Horten, Komphausbadstr. 10
5300 Bonn: Puppenkönig Westenhöfer, Gangolfstr. 8–10
5400 Koblenz: Komet, Carl-Zeiss-Str. 4
5500 Trier: Büro-Center Lehr, Güterstr. 82 · Horten, Fleischstr. 68–76
5600 Wuppertal: HiFi-Studio 9, Gewerbeschulstr. 9
5620 Velbert: Noethen KG, Kurze Str. 14
6000 Frankfurt: Hertie, Zeil 90 · Main Radio, Kaiserstr. 40 · Quelle, Zeil 46 Saturn Hansa, Bergerstr. 125
6080 Groß-Gerau: Van Venrooy, Gernsheimer Str. 1
6082 Mörfelden: Radio Landau, Langener Str. 8
6092 Kelsterbach: Radio Lippert, Lange Str. 22
6100 Darmstadt: Quelle, Ludwigsplatz 6
6113 Babenhausen: D. Sähring GmbH, H. d. Altdörfer Kirche
6143 Lorsch: Stereo-Haus, Hirschstr. 52
6200 Wiesbaden: Hertie, Kirchgasse 6
6230 Frankfurt: Main Radio (80), Main-Taunus-Zentrum
6300 Gießen: Horten, Bahnhofstr. 9
6400 Fulda: Media-Markt, Rangstr. 25
6450 Hanau: CP Computer Products, Altstr. 2–6
6460 Gelnhausen: Kaufhaus Joh, Im Ziegelhaus 2–4
6500 Mainz: Radio-Bauer GmbH & Co. KG (1), Ludwigstr. 3 · Quelle, Am Brand 41
6520 Worms: Horten, Am Römischen Kaiser 7 · Rheinelektra, Wilhelm-Leuschner-Str. 15
6630 Saarlouis: Shop 64, Titzstr./ Liesdorferstr.
6642 Orscholz: Radio - Fernseh, Bürotechnik, Peter Habermehl, Schmiedewäldchen 1
6650 Homburg: Shop 64, Talstr. 44
6680 Neunkirchen: Shop 64 Computerzentrum, Wellersweilerstr. 13
6688 Illingen: Radio-Schneider, Hauptstr. 28
6700 Ludwigshafen: Horten, Bismarckstr. 63
6750 Kaiserslautern: Radio Frohnhöfer, Rosenstr. 10 · Quelle, Marktstr. 15 · Spiel + Freizeit, Carl Gotthold, Marktstr. 7
6780 Pirmasens: Horten, Hauptstr. 13 Radio Bruckner GmbH, Schloss Str. 3
6800 Mannheim: Horten, N7 · Quelle, 07–13 · Rheinelektra, P7, 25
6806 Viernheim: Klaus Arnet, Rathausstr. 70
6822 Altlussheim: Radio Guido Siegmann, Rheinhäuserstr. 21
6900 Heidelberg: Horten, Bergheimer Straße 1
6980 Wertheim-Bettingen: Friedrich Lohmann, Wolfsgasse 8
7000 Stuttgart: Horten (1), Eberhard-Str. 28 · Radio Grüner, Marienstr. 3 Compec Weiss & Co., Königstr. 16
7100 Heilbronn: Horten, Fleiner Str. 15
7170 Schwäbisch Hall: Wolpert, Bahnhofstr. 11
7320 Göppingen: Rheinelektra, Freihofstr. 5/Am Bahnhof
7410 Reutlingen: Horten, Karlstr. 20
7475 Meßstetten: Heim- & PC-Computermarkt, Hauptstr. 10
7500 Karlsruhe: F. W. Döring, Ritterstr. 5–7 · Hertie, Kaiserstr. 92 · Quelle, Kaiserstr. 135 · Radio Freytag, Karlstr. 30-32
7530 Pforzheim: Horten, Westl. K.-Friedrich-Str. 17
7550 Rastatt: Computer- u. Bürotechnik T. Rieger, Bahnhofstr. 38
7580 Buehl: Rheinelektra, Hauptstr. 65
7590 Achern: Rheinelektra, Am Markt 11
7607 Neuried: Electro Müntzer GmbH (2), Friedrichstr. 16
7800 Freiburg: Hertie, Kaiser-Joseph-Str. 165
7900 Ulm: Horten, Bahnhofstr. 5 Computer Studio, Claus Wecker, Hafenbad 18/1
8000 München: Hertie, Bahnhofsplatz 7 Kaufhof AG, Am Stachus
8000 München: Computer-Center, Schillerstr. 17
8032 Gräfelfing: Pro TV+HiFi Markt, Pasingerstr. 94
8070 Ingolstadt: Dreyer GmbH, Manchinger Str. 125
8170 Bad Tölz: Bavaria HiFi, Bahnhofstr. 8
8390 Passau: Deuringer, Bahnhofstr. 15
8400 Regensburg: Horten, Neupfarrplatz 8
8500 Nürnberg: Horten (1), Aufseßplatz 18 · Video Magic, Gostenhofer Hauptstr. · Quelle, Fürther Str. 205
8662 Helmbrecht: Uhlig Büromaschinen GmbH, Hofer Str. 9
8670 Hof: Radio Granz, Königstr. 50
8700 Würzburg: ZE Electronic, Juliuspromenade
8707 Veitshöchheim: Radio Bauer GmbH, Bahnhofstr. 2
8720 Schweinfurt: Horten, Jägersbrunnen 11–13 · Zierhut Electronic, Carl-Zeiss-Str. 33
8735 Ebenhausen: Karl-Heinz Seufert, Schweinfurter Str. 5
8752 Mömbris: Menzel Computique, Im Kahltalzentrum
8758 Goldbach: Fernseh Hugo KG, Aschaffenburger Str. 16
8760 Miltenberg: Spielwaren Fuerst, Hauptstr. 157
8800 Ansbach: Radio Busch, Uzstr. 8–10

Otto Versand, 2000 Hamburg

Scan: S. Höltgen


```
1830 ST=ST+.5
1840 IF ST=360 THEN ST=0
1850 XH=SIN(ST*BO)*RH
1860 YH=COS(ST*BO)*RH
1870 NPLOT X0, Y0 TO X2, Y2
1880 NPLOT X0, Y0 TO X3, Y3
1890 GOSUB 1950
1900 POKE&H43A1, Z3
1910 POKE&H43A2, Z4
1920 CALL 43A0
1930 RETURN
1940 REM ----------- Zeiger plotten ------------
1950 FCOLOUR2
1960 PLOT X0, Y0 TO X0+XM, Y0-YM
1970 FCOLOUR3
1980 PLOT X0, Y0 TO X0+XH, Y0-YH
1990 FCOLOUR4
2000 RETURN
```

Turmspringen

Nach der Anfangsmelodie werden Sie nach der Anzahl der Mitspieler gefragt. Hiernach müssen Sie die jeweiligen Namen und Nationen eingeben. Nun erscheint das Spielbild. Sie stehen auf dem Sprungturm und müssen nach dem Absprung durch geschickte Drehungen im Flug die höchstmögliche Punktzahl erreichen. Die Athleten haben jeweils drei Versuche, die getrennt bewertet werden. Jeder Mitspieler wird über die Anzeigetafel aufgerufen. Hier sieht er auch die Bewertung der einzelnen Sprünge.

Variable	Bedeutung
F§(),K§(),Y§(), A§()	einzelne Sprungfiguren
F,K	zum Einlesen der Titelmelodie
ANZ	Anzahl der Spieler
NAM§()	Namen der Spieler
NAT§()	Nationen der Spieler
QI	Anzahl der Sprünge jedes Spielers
TT	Anzahl der Durchgänge
ZX	Abfrage, ob doppelter Salto
PU	allgemeine Variable der Punkte
PUM1,PUM2, PUM3	Variable der Zwischenpunktzahl
PUNKT()	Endpunktzahl jedes Spielers
M	Variable zur Umdefinierung der einzelnen Figuren
COL	Farbvariable
O	Länge des Applauses
W	Länge der Verzögerung
diverse Variablen	zum Aufbau des Bildschirmes und so weiter

Variablenliste

Steuerung:
Drücken des Feuerknopfes – Sie springen ab
Joystick nach rechts – Ihr Springer rollt sich zusammen
Joystick nach links – Ihr Springer knickt ein
Joystick nach unten – Ihr Springer streckt sich.
Sind alle Sprünge absolviert, so wird in der Auswertung die Plazierung der einzelnen Spieler gezeigt.
Die Summe der Sprungnoten wurden hierzu addiert.
Das Drücken einer beliebigen Taste beendet das Spiel.
Es erscheint nun noch die Abfrage, ob Sie noch einmal spielen möchten.

Tips:
Stechen Sie nach Möglichkeit gestreckt mit einem Kopfsprung ins Wasser ein, da es sonst Abzugspunkte gibt.
Führen Sie Drehungen bei Rollen nach Möglichkeit voll aus. Brechen Sie keinen Salto durch eine andere Figur ab.
Höchstpunktzahl:
10 Punkte
Mindestpunktzahl:
0,5 Punkte
Geben Sie Leerzeichen bei der Abfrage nach Namen und Nation immer durch „Shift – 2" ein.

Christoph Hinsch

```
10 CALL ANFANG
20 CALL LAUFSCHRIFT("INTERNATIONAL HIGH-
DIVING","CHAMPIONSHIP OF FRANCE")
30 CALL HYMNE :: CALL MAGNIFY(3):: CALL
CLEAR :: CALL SCREEN(2)
40 FOR COL=1 TO 14 :: CALL COLOR(COL,12,
2):: NEXT COL
50 ON WARNING NEXT
60 DISPLAY AT(21,1):"LEERZEICHEN ALS-@-E
INGEBEN !"
70 DISPLAY AT(23,1):"ANZAHL DER ATHLETEN
 (1-5)?" :: ACCEPT AT(23,28)BEEP SIZE(1)
VALIDATE("12345"):ANZ
80 FOR I=1 TO ANZ :: PRINT : :"NAME ATHL
.";I;":": :
90 ACCEPT AT(22,16)BEEP VALIDATE(UALPHA,
".@")SIZE(8):NAM$(I):: IF NAM$(I)="" THE
N 90
100 PRINT "NATION (KURZ):": :
110 ACCEPT AT(22,16)BEEP VALIDATE(UALPHA
,"@")SIZE(5):NAT$(I):: IF NAT$(I)="" THE
N 110
120 NEXT I
```

Scan: S. Höltgen


```
130 CALL CLEAR
140 GOSUB 490 :: CALL ZEICHENDEF(A$()):: 
 CALL AUFBAU
150 FOR TT=1 TO ANZ :: FOR QI=1 TO 3
160 M,PU,ZX=0 :: CALL TAFEL
170 CALL CHAR(104,A$(1)):: CALL SPRITE(#
1,104,2,31,119)
180 DISPLAY AT(3,9)SIZE(2):QI :: DISPLAY
 AT(3,9)SIZE(1):"@"
190 DISPLAY AT(3,1)BEEP SIZE(LEN(NAM$(TT
))):NAM$(TT):: DISPLAY AT(5,1)SIZE(LEN(N
AT$(TT))):NAT$(TT):: DISPLAY AT(5,7)SIZE
(4):"00.0"
200 CALL VERZ(100)
210 CALL KEY(1,K,S):: CALL JOYST(1,X1,Y1
)
220 IF K=18 THEN 460
230 GOTO 210 :: END
240 CALL KEY(1,K,S):: CALL JOYST(1,X1,Y1
)
250 IF X1=0 AND Y1=-4 THEN M=M+1 :: PU=P
U+.5 :: CALL CHAR(104,Y$(M)):: IF M>3 OR
 M<0 THEN M=0
260 IF X1=-4 AND Y1=0 THEN M=M+1 :: CALL
 CHAR(104,F$(M)):: IF M=2 THEN PU=PU+2.5
 ELSE IF M>3 THEN M=0
270 IF X1=4 AND Y1=0 THEN ZX=ZX+1 :: M=M
+1 :: CALL CHAR(104,K$(M)):: IF M=4 THEN
 PU=PU+3.5 :: M=0
280 CALL POSITION(#1,B,H):: IF B>154 THE
N 300
290 GOTO 240 :: END
300 CALL SOUND(-1500,-5,0):: CALL COLOR(
#1,8):: CALL PATTERN(#1,108):: CALL MOTI
ON(#1,1,-1)
310 CALL VERZ(500)
320 IF M=4 THEN PU=PU-4.5
330 IF M=3 THEN PU=PU-2.5
340 IF M=2 THEN PU=PU+.5
350 IF M=1 THEN PU=PU-3.5
360 IF ZX=9 THEN PU=PU+2
370 IF PU<1 THEN PU=.5 :: DISPLAY AT(5,9
)SIZE(2):USING ".#":PU :: DISPLAY AT(5,7
)SIZE(2):"@0" :: GOTO 390
380 IF PU<9.5 THEN DISPLAY AT(5,8)SIZE(3
):USING "#.#":PU :: DISPLAY AT(5,7)SIZE(
1):"@"
390 IF PU>10 OR PU=10 THEN PU=10 :: DISP
LAY AT(5,7)BEEP SIZE(4):USING "##.#":PU
400 CALL MOTION(#1,0,0):: CALL APPLAUS(P
U+2):: CALL DELSPRITE(#1)
410 CALL VERZ(1000)
420 IF QI=1 THEN PUM1(TT)=PU
430 IF QI=2 THEN PUM2(TT)=PU
440 IF QI=3 THEN PUM3(TT)=PU
450 NEXT QI :: NEXT TT :: GOTO 640 :: EN
D
460 CALL CHAR(104,Y$(1))
470 CALL MOTION(#1,-10,-10):: CALL VERZ(
50):: CALL SOUND(-100,-4,0):: CALL CHAR(
104,A$(2)):: CALL MOTION(#1,10,0):: CALL
 VERZ(20)
480 CALL CHAR(104,Y$(2)):: CALL MOTION(#
1,8,-4):: GOTO 240
490 A$(1)="404C6C3C1E0E0F07070F0F1E180E0
306"&RPT$("0",32)
500 A$(2)=RPT$("0",11)&"606E73F03010"&RP
T$("0",22)&"3F6FCF0E"&RPT$("0",11)
510 Y$(1)="0080C066361F0F030301"&RPT$("0
",24)&"80C0E0E0F070380C060C"
520 Y$(2)=RPT$("0",15)&"1030F0F0E1C3060C
00103060C3878F0E0E0C08"&RPT$("0",11)
530 Y$(3)="80C060301E0F0F0301"&RPT$("0",
24)&"80C0C0E0E07070780C0603"
540 Y$(4)=RPT$("0",11)&"30307070F0E0C183
060C00E1B3078F8F8E0C08"&RPT$("0",15)
550 F$(1)=RPT$("0",9)&"F"&RPT$("0",9)&"8
0F0F"&RPT$("0",12)&"707070F83C1C1C3CFCF8
"&RPT$("0",8)
560 F$(2)=RPT$("0",7)&"3073F3F3E3C040404
0404040000000E0F0F070303030303030307000
"
570 F$(3)="0000000F1F1E1C1C1C1E0F070707"
&RPT$("0",10)&"F8F80800000000F8"&RPT$("0
",10)
580 F$(4)="000000060C0C0C0C0C0E0F0F07"
&RPT$("0",10)&"20202020203838 78F8E0C0000
0"
590 K$(1)=RPT$("0",9)&"3434361371F390F07
"&RPT$("0",14)&"808080C0C0C0C0C08"&RPT$(
"0",7)
600 K$(2)="00000073F3F3F3B0301010101030E"
&RPT$("0",10)&"C0E0E0E06060C080C00000000
0"
610 K$(3)="0000000F1F1E1F1F1C0E0E0E"&RPT
$("0",14)&"4080C0E030101"&RPT$("0",13)
620 K$(4)="00000000203070F0D0C0E0F0F070
000000070C08000000080C0D8F8F8F8C00000"
630 RETURN :: END
640 CALL CLEAR :: CALL DELSPRITE(ALL)
650 FOR COL=1 TO 14 :: CALL COLOR(COL,12
,2):: NEXT COL
660 DISPLAY AT(2,4):"==== AUSWERTUNG ===
="
670 PUNKT(1)=PUM1(1)+PUM2(1)+PUM3(1)
680 PUNKT(2)=PUM1(2)+PUM2(2)+PUM3(2)
690 PUNKT(3)=PUM1(3)+PUM2(3)+PUM3(3)
700 PUNKT(4)=PUM1(4)+PUM2(4)+PUM3(4)
710 PUNKT(5)=PUM1(5)+PUM2(5)+PUM3(5)
720 FOR I=1 TO ANZ-1 :: FOR J=I+1 TO ANZ
730 IF PUNKT(I)>=PUNKT(J)THEN 750
740 Z=PUNKT(I):: PUNKT(I)=PUNKT(J):: Q$=
NAM$(I):: W$=NAT$(I):: NAM$(I)=NAM$(J)::
 NAT$(I)=NAT$(J):: NAM$(J)=Q$ :: NAT$(J)
=W$ :: PUNKT(J)=Z
750 NEXT J :: NEXT I
760 FOR I=1 TO ANZ
770 DISPLAY AT(3+3*I,1):I;".";" ";NAM$(I
),NAT$(I)
780 DISPLAY AT(3+3*I,22):PUNKT(I)
790 NEXT I
800 DISPLAY AT(24,9):"TI-COMPUTER"
810 CALL KEY(0,K,S):: IF S=0 THEN 810
820 CALL CLEAR :: DISPLAY AT(12,1):"NOCH
 EINEN DURCHGANG ?(J/N)"
830 CALL KEY(0,K,S)
840 IF K=78 OR K=110 THEN CALL CLEAR ::
END
850 IF K=74 OR K=106 THEN 130
860 GOTO 830 :: END
870 SUB ANFANG :: CALL CLEAR
880 CALL CHAR(123,RPT$("F",16),129,RPT$(
"F",16),130,"0103070F1F3F7FFF",131,"80C0
E0F0F8FCFEFF",132,RPT$("F0",8))
890 CALL CHAR(96,RPT$("F",64),100,RPT$("
01",16)&RPT$("F",32))
900 CALL MAGNIFY(4):: CALL SCREEN(2):: F
OR COL=1 TO 14 :: CALL COLOR(COL,2,2)::
NEXT COL
910 CALL HCHAR(21,1,123,128)
920 SA=13 :: ZA=11 :: DA=12
930 FOR I=1 TO 7 :: CALL HCHAR(SA,ZA,129
,DA):: SA=SA+1 :: ZA=ZA-1 :: DA=DA+2 ::
NEXT I
940 CALL HCHAR(20,4,129,25):: CALL VCHAR
(4,4,132,16)
950 LA=13 :: KA=11 :: FOR I=1 TO 7 :: CA
LL HCHAR(LA,KA,130,1):: LA=LA+1 :: KA=KA
-1 :: NEXT I
960 FA=13 :: JA=22 :: FOR I=1 TO 7 :: CA
```

Scan: S. Höltgen

```
LL HCHAR(FA,JA,131):: FA=FA+1 :: JA=JA+1
 :: NEXT I
970 NA=21 :: FOR I=1 TO 4 :: CALL HCHAR(
NA,1,129,5):: CALL HCHAR(NA,28,129,5)::
NA=NA+1 :: NEXT I
980 FOR H=16 TO 17 :: CALL VCHAR(21,H,12
9,4):: NEXT H
990 FOR COL=1 TO 14 :: CALL COLOR(COL,2,
8):: NEXT COL
1000 CALL SCREEN(8):: CALL COLOR(12,10,8
,13,2,8,1,8,8)
1010 DISPLAY AT(12,11)SIZE(8):"LE STADE"
 :: SUBEND
1020 SUB LAUFSCHRIFT(Q$,P$)
1030 FOR I=1 TO 12 :: CALL VERZ(50):: DI
SPLAY AT(1,24-2*I)SIZE(4+2*I):Q$ :: NEXT
 I
1040 FOR I=1 TO 18 :: DISPLAY AT(3,24-I)
SIZE(4+2*I):P$ :: CALL VERZ(30):: NEXT I
1050 SUBEND
1060 SUB HYMNE
1070 CALL SPRITE(#1,100,5,120,15,-5,0,#2
,96,16,120,33,-5,0,#3,96,9,120,51,-5,0)
1080 CALL VERZ(1500)
1090 CALL MOTION(#1,0,0,#2,0,0,#3,0,0)
1100 DATA 63,147,189,147,63,147,250,196,
250,196,250,220,250,220,250,294,250,247,
189,196,63,196,189,247,63,196
1110 DATA 250,165,500,262,189,220,63,185
,500,196,250,196,125,30000,189,196,63,22
0,250,247,250,247,250,247,189,262
1120 DATA 63,247,125,247,125,220,500,220
,189,220,63,247,250,262,250,262,250,262,
189,294,63,262,500,247
1130 DATA 250,30000,189,294,63,294,250,2
94,189,247,63,196,250,294,189,247,63,196
,500,147,189,30000,63,147
1140 DATA 125,147,125,185,500,220,250,26
2,125,220,125,185,250,196,250,196,500,17
5,250,165
1150 DATA 125,196,125,196,250,196,125,18
5,125,196,500,220,250,30000,250,220,375,
233,125,247,125,220,125,247,125,262
1160 DATA 125,294,500,220,250,30000,125,
233,125,220,375,196,125,196,125,196,125,
233,125,220,125,196,125,196
1170 DATA 125,185,250,185,565,30000,63,2
94,500,294,189,294,63,294,189,247,63,196
,500,220,125,30000,63,294
1180 DATA 500,294,189,294,63,294,189,247
,63,196,500,220,125,30000
1190 DATA 125,30000,250,147,750,196,250,
220,750,247,250,30000,500,262,250,294,25
0,330,750,220,250,330
1200 DATA 500,294,189,294,63,247,189,262
,63,220,500,196,125,196,3000,30000
1210 FOR I=1 TO 123 :: READ F,K :: CALL
SOUND(F,K,0):: NEXT I
1220 CALL DELSPRITE(#1,#2,#3):: SUBEND
1230 SUB ZEICHENDEF(A$())
1240 CALL CHAR(136,"FF0000FFFFFF0000",13
7,"F80402F1F8FC0E07")
1250 CALL CHAR(138,"0000000080402010",13
9,"")
1260 CALL CHAR(140,"0301",141,"88C4E2713
81C0E07")
1270 CALL CHAR(129,RPT$("F",16))
1280 CALL CHAR(37,"F0180C060202FE02",131
,RPT$("02",6)&"FF")
1290 CALL CHAR(132,"894423118844 42D1",64
,"")
1300 CALL CHAR(133,"0072700B63670099",13
4,RPT$("2F",8))
1310 CALL CHAR(38,RPT$("0",12)&"FFFF",39
,"FF"&RPT$("80",5)&"FFFF",112,"")
1320 CALL CHAR(96,"FF7F3F1F0F070301"&RPT
$("0",16)&RPT$("F",18)&"FF3F1F0F070301")
1330 CALL CHAR(100,"0000000E1A3222222020
20203F3F2000000000003868C888888080808080
8000")
1340 CALL CHAR(104,A$(1))
1350 CALL CHAR(108,"0200041100004801012 5
0000100144000040000820820 8C0CAC004002800
0210")
1360 SUBEND
1370 SUB AUFBAU
1380 L=6 :: Z=18 :: S=15 :: Q=16 :: A=18
 :: W=18 :: M=17 :: T=19 :: E=15 :: H=6
:: O=121 :: R=113 :: P,C=9 :: D=20 :: B=
2
1390 CALL CLEAR :: CALL SCREEN(2)
1400 FOR COL=1 TO 14 :: CALL COLOR(COL,2
,2):: NEXT COL
1410 CALL HCHAR(14,1,112,352)
1420 CALL HCHAR(9,1,133,30):: CALL HCHAR
(10,1,133,31):: CALL HCHAR(11,1,133,64)
1430 FOR I=1 TO 5 :: CALL HCHAR(B,2,64,1
2):: B=B+1 :: NEXT I
1440 CALL VCHAR(7,4,134,2):: CALL VCHAR(
7,11,134,2)
1450 FOR I=1 TO 4 :: CALL HCHAR(C,D,132)
:: C=C+1 :: D=D+1 :: NEXT I
1460 FOR I=1 TO 4 :: CALL HCHAR(P,L,132)
:: P=P+1 :: L=L+1 :: NEXT I
1470 CALL HCHAR(17,1,136,14):: CALL HCHA
R(17,15,137):: CALL HCHAR(17,16,138):: C
ALL HCHAR(13,1,136,32)
1480 CALL HCHAR(6,16,38,4):: CALL HCHAR(
6,20,39,9):: CALL HCHAR(6,29,37):: CALL
VCHAR(7,29,131,14)
1490 CALL HCHAR(7,21,129,8)
1500 FOR I=8 TO 20 :: CALL HCHAR(I,27,12
9,2):: NEXT I :: CALL HCHAR(8,25,129)
1510 CALL VCHAR(9,26,134,12):: CALL HCHA
R(8,26,129)
1520 FOR I=1 TO 14 :: CALL VCHAR(18,I,13
9,7):: NEXT I
1530 FOR I=1 TO 7
1540 CALL HCHAR(Z,S,140):: S=S+1 :: Z=Z+
1
1550 CALL HCHAR(A,Q,141):: A=A+1 :: Q=Q+
1
1560 CALL HCHAR(W,M,138):: W=W+1 :: M=M+
1
1570 NEXT I
1580 FOR I=1 TO 6 :: CALL VCHAR(T,E,139,
H):: T=T+1 :: E=E+1 :: H=H-1 :: NEXT I
1590 FOR I=24 TO 28 :: CALL SPRITE(#I,96
,10,O,R):: O=O+16 :: R=R+16 :: NEXT I
1600 CALL SPRITE(#20,96,8,64,233,#21,96,
8,72,241)
1610 CALL SPRITE(#23,100,2,122,60)
1620 FOR COL=1 TO 14 :: CALL COLOR(COL,1
2,2):: NEXT COL
1630 CALL COLOR(1,2,8,14,2,5,11,2,10,6,1
2,2,7,12,2,13,2,10,5,12,2)
1640 SUBEND
1650 SUB APPLAUS(O)
1660 FOR LAENGE=1 TO O+7
1670 CALL SOUND(-1000,-6,5,900,10)
1680 CALL CHAR(133,"F8727389FD708E03",13
3,"0072700B63670099")
1690 NEXT LAENGE
1700 SUBEND
1710 SUB VERZ(W):: FOR I=1 TO W :: NEXT
I :: SUBEND
1720 SUB TAFEL :: B=2
1730 FOR I=1 TO 5 :: CALL HCHAR(B,2,64,1
2):: B=B+1 :: NEXT I
1740 SUBEND
```

Scan: S. Höltgen

Stichwort: Home-Computer

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

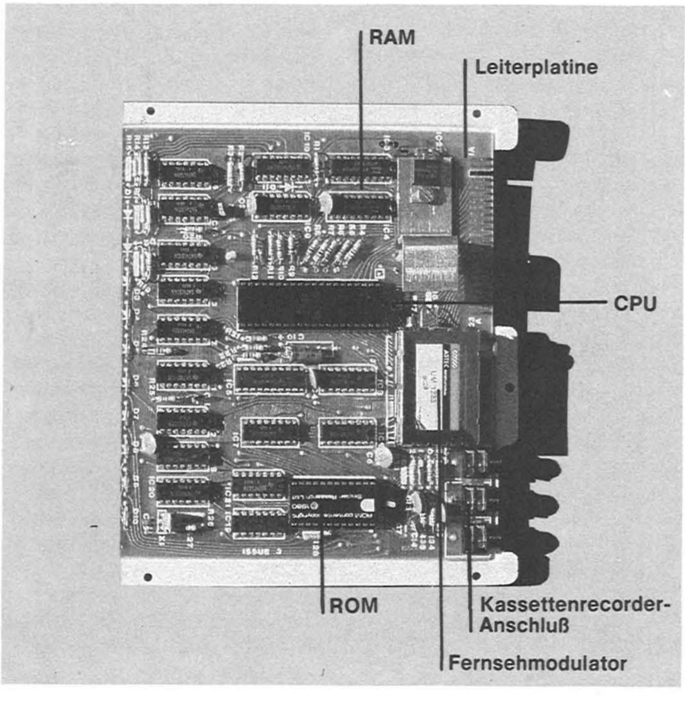
Im ersten Stichwort steht der Home-Computer oder, wie man in der Großdatenverarbeitung sagen würde, die Zentraleinheit im Mittelpunkt. Die Zentraleinheit ist das Herz des Home-Computers, die alle Funktionen steuert.

○ **Leiterplatinen** sind die Plattform, auf der der Home-Computer aufgebaut ist. Sie besteht aus Isoliermaterial (Fiberglas oder Pertinax). Leiterbahnen aus einer dünnen Metallschicht verbinden die elektronischen Bauteile. Die auf der Leiterplatte untergebrachten Hauptkomponenten sind die CPU, der ROM-Speicher, der RAM-Speicher und der Fernsehmodulator.

○ **CPU** ist die Zentraleinheit (central processing unit), die alle Abläufe im Home-Computer steuert. So sorgt die CPU dafür, daß die entweder direkt oder als Programm eingegebenen Befehle richtig ausgeführt werden. Gesteuert werden die Signale im Takt der Uhr (Timer), die ebenfalls Teil der CPU ist und mit einer Frequenz zwischen 2 und 4 MHz schwingt.

○ **ROM** ist der Lesespeicher des Mikrocomputers. Im ROM sind die Programme, die den Rechner steuern, gespeichert. Sie sind folgendermaßen unterteilt:

○ **Betriebssysteme** sorgen dafür, daß Zeichen von der Tastatur in den Rechner übernommen werden, daß diese Zeichen auf dem Monitor (in der Regel ein Farbfernsehgerät) dargestellt werden und daß Programme ausgeführt und auf Kassette oder Diskette ab-



Scan: S. Höltgen

gespeichert und gelesen werden können.

○ **RAM** ist der Schreib-Lese-Speicher, in den Informationen geschrieben und aus dem Informationen gelesen werden können. Er wird gelöscht, wenn die Stromversorgung abgeschaltet wird. Eine Ausnahme sind CMOS-Speicher. Sie speichern die Informationen auch dann, wenn sie über einen Akku oder eine Batterie mit Strom versorgt werden.

○ **RAM-Speicher** werden von der CPU und dem Betriebssystem in verschiedene Abschnitte aufgeteilt:

○ **System-Variable** enthalten Informationen zur Systemverwaltung, zum Beispiel, an welche Stelle das nächste Zeichen am Bildschirm geschrieben wird.

○ **Programm-Bereich** ist der Teil des Arbeitsspeichers, in dem Programme gespeichert werden.

○ **Bildschirm-Speicher** sind jene Teile des Speichers, in dem eine Kopie dessen abgelegt ist, was auf dem Bildschirm dargestellt wird.

○ **Variablen-Speicher** speichern die Daten, die den Variablen im Programm zugeordnet sind.

○ **Statuszeilen** speichern die Informationen ab, die gerade eingegeben werden.

○ **Rechen-Stack** ist ein Stapelspeicher, mit dessen Hilfe die CPU arithmetische Funktionen ausführt.

○ **Reservierter Bereich** ist ein Speicherabschnitt, der von der CPU als Arbeitsspeicher genutzt werden kann, der aber erst dann zur Verfügung steht, wenn Speichererweiterungen in den Rechner eingebaut werden.

○ **Maschinen-Stack** ist der Bereich, wo interne Informationen für die Arbeit der CPU abgelegt werden, wie zum Beispiel die Zeilennummern, die in einem Programm vorkommen.

○ **GOSUB-Stack** hält die Zeilennummer fest, zu der zurückgesprungen werden muß, wenn ein Unterprogramm abgearbeitet ist.

○ **Dienstprogramme** erleichtern den Umgang mit dem Home-Computer.

○ **Editor** ist ein Hilfsprogramm, mit dem Programme in den Rechner eingegeben werden können. Viele Home-Computer verfügen nur über einen zeilenorientierten Editor, mit dem eine Programmzeile bereitgestellt wird, in der Zeichen gelöscht, eingefügt oder geändert werden können. Andere Rechner, wie der VC 20, verfügen über einen Bildschirmeditor, der einen ganzen Abschnitt des Programms anzeigt, in dem geändert werden kann.

○ **Monitorprogramme** überwachen unter anderem die Tastatur. Wird eine Taste gedrückt, wird der Taste ein Wert zugeordnet, der dem Rechner übergeben wird. Dieser Wert ist in der Regel ein ASCII-Code (American Standard for Information Interchange).

○ **BASIC-Interpreter** übersetzen Befehle der Programmiersprache BASIC (Beginners all Purpose Symbolic Instruction Code) zeilenweise in maschinenausführbare Form, die Zeile für Zeile ausgeführt wird. Der Vorteil ist, daß formale Fehler leicht geändert werden können. Für logische Fehler ist allerdings der Programmierer zuständig.

○ **Kassettenrecorder-Schnittstelle** ist der Aus- und Eingang des Home-Computers, an dem digitale Information (Programme und Daten) in analoge Impulse übersetzt werden oder umgekehrt, um Information auf Magnetband zu speichern. Leider speichert jeder Home-Computer die Information nach einer anderen Norm, so daß Programme vom Hersteller S nicht vom Computer des Herstellers C gelesen werden können und umgekehrt.

○ **Schnittstelle** ist die Verbindung zur Außenwelt, um andere Geräte oder Baugruppen an den Computer anzuschließen. Die meisten Home-Computer verfügen über besondere Stecker, um Kassettenrecorder, Diskettenlaufwerke, Speichererweiterungen, Drucker, Lichtgriffel oder Grafiktabletts anzuschließen. Andere verfügen über einen Standardanschluß, an dem die Erweiterungen übereinandergesteckt werden.

Scan: S. Höltgen

Stichwort: Programmieren

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

Programmieren fängt, wie man zunächst denkt, nicht mit dem Programm an, sondern mit dem Problem.

Problemanalyse ist der Vorgang, der immer am Anfang steht. Die Aufgabe, gleichgültig, ob es sich um ein Spiel oder um ein Rechenprogramm handelt, muß genau beschrieben sein und in viele kleine Stücke zerlegt werden. Um zu genauen Ergebnissen zu kommen, müssen alle verschiedenen Fälle, die in dem Programm auftreten können, genau bedacht sein. Wer mit einem einfachen Programm zwei Zeilen durcheinanderdividieren will, kann dem Rechner nicht einfach sagen: Ergebnis = Zähler geteilt durch Nenner, vielmehr muß in dem Programm auch die Möglichkeit berücksichtigt sein, die eintritt, wenn der Nenner gleich 0 wird.

Programmieren heißt, dem Computer in der richtigen Reihenfolge die Befehle mitzuteilen, die notwendig sind, um das Problem zu lösen. Dazu muß festgelegt werden, welche Informationen der Rechner benötigt, was er mit ihnen machen soll und wie er das Ergebnis seiner Arbeit mitteilen soll.

Um bei dem einfachen Rechenbeispiel zu bleiben: der erste Schritt wäre, die Abfrage von Zähler und Nenner über den Bildschirm, die Eingabe der Werte über die Tastatur, die der Zähler und Nenner annehmen soll. Der zweite Schritt ist ein Test, ob der Nenner gleich 0 ist. Wenn ja, muß die Nachricht ausgegeben werden: Diese Operation ist nicht erlaubt. In diesem Fall beginnt das Programm wieder beim ersten Schritt. Der dritte Schritt ist die Ausführung der Division und zum Schluß als vierter Schritt die Ausgabe des Ergebnisses auf dem Bildschirm oder dem Drucker.

Bei diesem Beispiel war bis jetzt noch keine Rede von Programmiersprachen. Denn die Aufgabe bis hierher war: Analyse des Problems und Beschreibung des Lösungsweges.

Die Hauptaufgabe beim Programmieren ist also die Beschreibung der Fragestellung und die Skizze des Lösungsweges, wobei alle auftretenden Fälle vorausgesehen werden sollten.

Programmablaufplan heißt eine Methode, um das Ergebnis der bisherigen Überlegungen übersichtlich darzustellen. Mit bestimmten Symbolen wird der Ablauf des Programms festgelegt und gleichzeitig bestimmt, was mit den zu verarbeitenden Daten geschehen soll. Die Informatiker reden anstelle eines Programmablaufplans auch von einem Struktogramm oder einem Datenflußplan.

Die Übersetzer dieser Pläne in ein Programm, das vom Computer ausgeführt werden soll, ist eine reine Übersetzungsaufgabe. So wie ein Text vom Englischen ins Deutsche übersetzt wird, wird der Lösungsweg in eine Computersprache übertragen.

Die Befehlsworte in Programmiersprache, wie zum Beispiel BASIC, Pascal oder Logo, sind daher Worte aus der englischen Umgangs- oder Fachsprache. Der deutsche Computerneuling, der in der Regel nicht über perfekte

Scan: S. Höltgen

Englischkenntnisse verfügt, steht zunächst in doppelter Hinsicht vor der Sprachbarriere. Die Begriffe sind nicht nur neu, sie kommen zudem aus der Fremdsprache Englisch.

Zu diesen Problemen kommen noch die eigenen Rechenstrukturen der Computer. Das fängt schom beim Zählen an. Der Mensch rechnet im Zehnersystem. Der Computer kennt nur die binäre Welt, also nur Nullen und Einser. Somit muß der Rechner die eingegebenen Zahlen zuerst ins Dualsystem übertragen.

BASIC-Interpreter stehen auch dem Home-Computer zur Programmausführung zur Verfügung. Interpreter heißt, daß das Programm Zeile für Zeile in eine für den Computer verständliche Sprache übersetzt wird. Für Ordnung in BASIC-Programmen sorgen Zeilennummern, die dem Interpreter sagen, in welcher Reihenfolge die Befehle ausgeführt werden sollen.

Texteditor ist ein Hilfsprogramm, um das Programm über Tastatur einzugeben und auf dem Bildschirm darzustellen. Die unterschiedlichen Computersysteme verfügen über verschiedene Texteditoren. Die einfachen Systeme wie der ZX81 erlauben nur die Korrektur ganzer Zeilen. Bei komfortableren Systemen kann innerhalb einer Zeile korrigiert werden.

Bildschirmeditor ist das komfortabelste Hilfsprogramm zum Eingeben und Verändern von Programmen. Das Programm wird auf dem Bildschirm dargestellt, und eine Lichtmarke, der Cursor, läßt sich mit vier Steuertasten an jede beliebige Stelle bewegen, an der dann Zeichen gelöscht oder eingefügt werden können.

Programme sind eine Folge von Befehlen, die der Computer der Reihe nach ausführt.

Programmiersprachen stellen Befehle zur Verfügung, um die unterschiedlichsten Aktionen am Computer zu veranlassen. Es gibt Befehle, um Informationen in den Rechner einzugeben, sie dort zu speichern, zu verarbeiten und wieder auszugeben.

Maschinensprache ist die Ebene, die mit den Befehlen PEEK und POKE erreicht werden kann. Die Programmierung erfolgt über Hexadezimalzahlen. Der Umgang mit diesen beiden Befehlen erfordert allerdings Detailkenntnisse über das Computersystem.

Dokumentation ist ein wichtiger Bestandteil beim Programmieren. In der Dokumentation wird festgehalten, wie das Programm arbeitet und welchen Zweck die einzelnen Befehle erfüllen. Dies kann sowohl in schriftlicher Form als auch in Kommentaren, die in das Programm eingefügt werden, geschehen. Auf diese Weise wird sichergestellt, daß auch andere Personen die Arbeitsweise des Programms verstehen und in der Lage sind, eventuelle Änderungen des Programms vorzunehmen.

Grundsätzlich gilt aber auch fürs Programmieren: grau ist alle Theorie. Das Programmieren von Home-Computern lernt man am besten, indem man mit Hilfe der Bedienungsanleitung einige Beispiele selbst eingibt. Nach diesem Anfangserfolg fällt es leicht, durch Ändern der ersten Programme immer mehr mit der Fremdsprache Programmiersprache vertraut zu werden.

Die verschiedenen Zahlsysteme beim Computer

Dezimal	Hexadezimal	Binär
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111
16	10	10000

Scan: S. Höltgen

Englischkenntnisse verfügt, steht zunächst in doppelter Hinsicht vor der Sprachbarriere. Die Begriffe sind nicht nur neu, sie kommen zudem aus der Fremdsprache Englisch.

Zu diesen Problemen kommen noch die eigenen Rechenstrukturen der Computer. Das fängt schom beim Zählen an. Der Mensch rechnet im Zehnersystem. Der Computer kennt nur die binäre Welt, also nur Nullen und Einser. Somit muß der Rechner die eingegebenen Zahlen zuerst ins Dualsystem übertragen.

BASIC-Interpreter stehen auch dem Home-Computer zur Programmausführung zur Verfügung. Interpreter heißt, daß das Programm Zeile für Zeile in eine für den Computer verständliche Sprache übersetzt wird. Für Ordnung in BASIC-Programmen sorgen Zeilennummern, die dem Interpreter sagen, in welcher Reihenfolge die Befehle ausgeführt werden sollen.

Texteditor ist ein Hilfsprogramm, um das Programm über Tastatur einzugeben und auf dem Bildschirm darzustellen. Die unterschiedlichen Computersysteme verfügen über verschiedene Texteditoren. Die einfachen Systeme wie der ZX81 erlauben nur die Korrektur ganzer Zeilen. Bei komfortableren Systemen kann innerhalb einer Zeile korrigiert werden.

Bildschirmeditor ist das komfortabelste Hilfsprogramm zum Eingeben und Verändern von Programmen. Das Programm wird auf dem Bildschirm dargestellt, und eine Lichtmarke, der Cursor, läßt sich mit vier Steuertasten an jede beliebige Stelle bewegen, an der dann Zeichen gelöscht oder eingefügt werden können.

Programme sind eine Folge von Befehlen, die der Computer der Reihe nach ausführt.

Programmiersprachen stellen Befehle zur Verfügung, um die unterschiedlichsten Aktionen am Computer zu veranlassen. Es gibt Befehle, um Informationen in den Rechner einzugeben, sie dort zu speichern, zu verarbeiten und wieder auszugeben.

Maschinensprache ist die Ebene, die mit den Befehlen PEEK und POKE erreicht werden kann. Die Programmierung erfolgt über Hexadezimalzahlen. Der Umgang mit diesen beiden Befehlen erfordert allerdings Detailkenntnisse über das Computersystem.

Dokumentation ist ein wichtiger Bestandteil beim Programmieren. In der Dokumentation wird festgehalten, wie das Programm arbeitet und welchen Zweck die einzelnen Befehle erfüllen. Dies kann sowohl in schriftlicher Form als auch in Kommentaren, die in das Programm eingefügt werden, geschehen. Auf diese Weise wird sichergestellt, daß auch andere Personen die Arbeitsweise des Programms verstehen und in der Lage sind, eventuelle Änderungen des Programms vorzunehmen.

Grundsätzlich gilt aber auch fürs Programmieren: grau ist alle Theorie. Das Programmieren von Home-Computern lernt man am besten, indem man mit Hilfe der Bedienungsanleitung einige Beispiele selbst eingibt. Nach diesem Anfangserfolg fällt es leicht, durch Ändern der ersten Programme immer mehr mit der Fremdsprache Programmiersprache vertraut zu werden.

Die verschiedenen Zahlsysteme beim Computer

Dezimal	Hexadezimal	Binär
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111
16	10	10000

Scan: S. Höltgen

Stichwort: Programmieren

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

Programmieren fängt, wie man zunächst denkt, nicht mit dem Programm an, sondern mit dem Problem.

Problemanalyse ist der Vorgang, der immer am Anfang steht. Die Aufgabe, gleichgültig, ob es sich um ein Spiel oder um ein Rechenprogramm handelt, muß genau beschrieben sein und in viele kleine Stücke zerlegt werden. Um zu genauen Ergebnissen zu kommen, müssen alle verschiedenen Fälle, die in dem Programm auftreten können, genau bedacht sein. Wer mit einem einfachen Programm zwei Zeilen durcheinanderdividieren will, kann dem Rechner nicht einfach sagen: Ergebnis = Zähler geteilt durch Nenner, vielmehr muß in dem Programm auch die Möglichkeit berücksichtigt sein, die eintritt, wenn der Nenner gleich 0 wird.

Programmieren heißt, dem Computer in der richtigen Reihenfolge die Befehle mitzuteilen, die notwendig sind, um das Problem zu lösen. Dazu muß festgelegt werden, welche Informationen der Rechner benötigt, was er mit ihnen machen soll und wie er das Ergebnis seiner Arbeit mitteilen soll.

Um bei dem einfachen Rechenbeispiel zu bleiben: der erste Schritt wäre, die Abfrage von Zähler und Nenner über den Bildschirm, die Eingabe der Werte über die Tastatur, die der Zähler und Nenner annehmen soll. Der zweite Schritt ist ein Test, ob der Nenner gleich 0 ist. Wenn ja, muß die Nachricht ausgegeben werden: Diese Operation ist nicht erlaubt. In diesem Fall beginnt das Programm wieder beim ersten Schritt. Der dritte Schritt ist die Ausführung der Division und zum Schluß als vierter Schritt die Ausgabe des Ergebnisses auf dem Bildschirm oder dem Drucker.

Bei diesem Beispiel war bis jetzt noch keine Rede von Programmiersprachen. Denn die Aufgabe bis hierher war: Analyse des Problems und Beschreibung des Lösungsweges.

Die Hauptaufgabe beim Programmieren ist also die Beschreibung der Fragestellung und die Skizze des Lösungsweges, wobei alle auftretenden Fälle vorausgesehen werden sollten.

Programmablaufplan heißt eine Methode, um das Ergebnis der bisherigen Überlegungen übersichtlich darzustellen. Mit bestimmten Symbolen wird der Ablauf des Programms festgelegt und gleichzeitig bestimmt, was mit den zu verarbeitenden Daten geschehen soll. Die Informatiker reden anstelle eines Programmablaufplans auch von einem Struktogramm oder einem Datenflußplan.

Die Übersetzer dieser Pläne in ein Programm, das vom Computer ausgeführt werden soll, ist eine reine Übersetzungsaufgabe. So wie ein Text vom Englischen ins Deutsche übersetzt wird, wird der Lösungsweg in eine Computersprache übertragen.

Die Befehlsworte in Programmiersprache, wie zum Beispiel BASIC, Pascal oder Logo, sind daher Worte aus der englischen Umgangs- oder Fachsprache. Der deutsche Computerneuling, der in der Regel nicht über perfekte

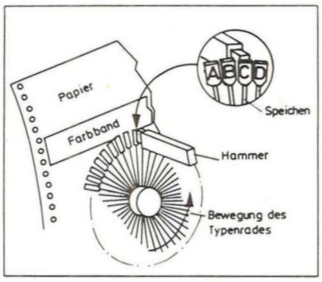
Scan: S. Höltgen

ander abgestimmt sind. Manche Matrixdrucker verfügen über Einzelpunktansteuerung. Dadurch wird es möglich, selbst programmierte Zeichen zu drucken oder in dieser Betriebsart Bilder zu erzeugen. Manche Home-Computer verfügen über Funktionen, mit denen der Bildschirminhalt, sei es Text oder Grafik, auf den Drucker ausgegeben werden kann (Hardcopy).

Die Technik, mit der Matrixdrucker die Zeichen erzeugen, wird immer ausgefeilter. Heute verfügen auch Matrixdrucker der unteren Preisklasse über mehrere Schrifttypen und Schriftgrößen, die programmgesteuert ausgewählt werden. Die Technik ist weitergeschritten und dadurch, daß die Nadeln leicht versetzt anschlagen, werden auch die „Löcher" zwischen den einzelnen Druckpunkten gefüllt.

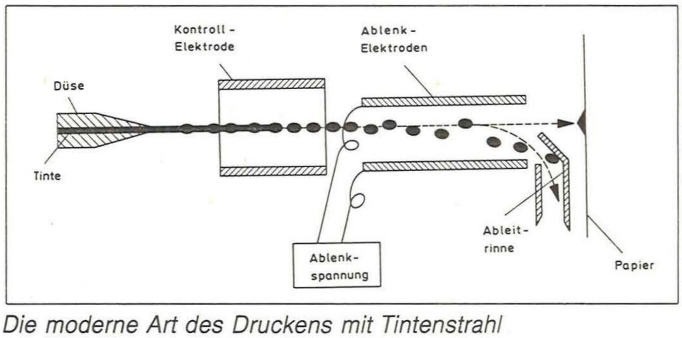
Moderne Matrixdrucker verfügen über bis zu 24 Nadeln, und ihr Druckbild ist auf den ersten Blick von dem der klassischen Schreibmaschine kaum zu unterscheiden. Techniken, die bislang teuren Druckern vorbehalten waren, sind inzwischen auch mit Geräten unter der 2000-Mark-Schwelle erreichbar. Es gibt bereits Matrixdrucker, die mit Hilfe eines vierfarbigen Farbbandes auch in verschiedenen Farben drukken. Daneben stehen Tintenstrahldrucker, die fast geräuschlos arbeiten. Auch Typenraddrucker, diese Technik löst bei der Schreibmaschine Typenhebel und Kugelkopf ab, sind inzwischen mit preiswerten Modellen am Markt.

Typenraddrucker besitzen ein Speichenrad, das an seinen Enden die erhabenen Buchstaben trägt, die in Druckposition geschwenkt und mit einem kleinen Hammer angeschlagen werden. Auf einem Typenrad können 86 oder 92 Zeichen untergebracht werden. Durch den Austausch des Rades können andere Schrifttypen und -größen eingesetzt werden. Die preiswerten Typenraddrucker erreichen allerdings nur Druckgeschwindigkeiten zwischen 10 und 20 Zeichen je Sekunde.



So funktioniert ein Typenrad-Drucker

Tintenstrahldrucker spritzen durch eine feine Düse winzige Tintentröpfchen auf das Papier. Sie werden durch Elektroden abgelenkt. Es war schwierig, eine geeignete Tinte zu finden, denn diese muß elektrisch aufladbar und so viskos sein, daß sie als rundes Tröpfchen auf dem Papier aufschlägt. Die technologischen Geschwindigkeiten werden deutlich, wenn man bedenkt, daß die Tintentropfen einen Durchmesser von 0,02 Millimeter haben, mit einer Geschwindigkeit von rund 700 km/h fliegen und beim Aufprall auf das Papier einen Fleck von 0,16 Millimeter erzeugen. Diese Drucker arbeiten schneller als preiswerte Matrixdrucker.



Die moderne Art des Druckens mit Tintenstrahl

Scan: S. Höltgen

Stichwort: Drucker

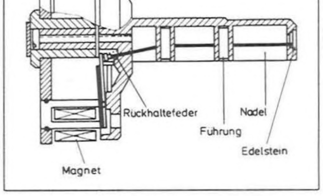
Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

Ein Drucker ist in der Regel das erste Peripheriegerät, das neben Speichererweiterung, Kassettenrecorder oder Diskettenlaufwerk für einen Home-Computer gekauft wird. Alle preiswerten Drucker arbeiten mit dem „Impact-dot-matrix"-Prinzip. Die Buchstaben und Zeichen werden aus einzelnen Punkten aufgebaut, die in einer 7*7- oder 7*9-Matrix das Zeichen abbilden.

Für einige Rechner, wie den HX-20, werden eingebaute Drucker angeboten, die als Thermodrucker nach dem gleichen Prinzip arbeiten. Der Sinclair-Drucker arbeitet nach dem Elektroerosionsprinzip und baut die Zeichen ebenfalls in einer Matrix auf.

Matrixdrucker verfügen über einen Druckkopf mit sieben oder neun Nadeln, die eng übereinander in einer Reihe stehen. Der eingebaute Zeichengenerator steuert den „Abschuß" der Nadeln, um ein Zeichen zu drucken. Die Nadeln drücken das Farbband auf das Papier und setzen somit aus lauter einzelnen Punkten ein Zeichen zusammen.

Die Elektronik des Druckers erzeugt nicht nur die Punktmuster, sondern sie „weiß" auch beim Druck, wo der Druckkopf



Schnittbild eines Matrix-Druckkopfs

auf der Zeile steht und steuert ihn auf Zehntelmillimeter genau. Der Druckvorgang läuft mit einer Geschwindigkeit von 80 bis 200 Zeichen pro Sekunde. Bei einem langsamen Drucker bedeutet dies, daß in einer Sekunde eine Zeile gedruckt wird, da der Druckkopf in der Sekunde 560mal positioniert wird und bis zu 5000mal je Sekunde die Matrixnadeln zum Aufbau der Zeichen angeschlagen werden.

Die Elektronik des Druckers überwacht gleichzeitig den Datenaustausch mit dem Home-Computer. Da eine ganze Reihe von schnell ablaufenden Vorgängen überwacht und gesteuert werden muß, verfügen fast alle Matrixdrucker über einen eigenen Mikroprozessor.

Je nach Druckermodell bieten die einzelnen Hersteller auch bereits bei preiswerten Druckern mehr Möglichkeiten als nur die 26 Buchstaben des Alphabets in Groß- und Kleinschreibung und die 10 Ziffern darzustellen. Die Datenübertragung zwischen Home-Computer und Drucker erfolgt im 8-Bit-ASCII-Code (ein amerikanischer Standard zum Datenaustausch), der 256 verschiedene Zeichen darstellen kann. Zieht man davon rund zwei Dutzend Steuerzeichen ab, bleiben immer noch mehr als 200 codierbare Zeichen übrig. Die rund 100 freien Zeichenplätze, die nicht zur Darstellung von Texten benötigt werden, sind in der Regel mit Grafikzeichen belegt. Diese Grafikzeichen entsprechen genau denen des Home-Computers, wenn der Home-Computer über Block-Grafik verfügt und Computer und Drucker aufein-

Scan: S. Höltgen

werden. Der eigentliche Datensatz besteht, je nachdem, ob sieben oder acht bit ASCII gesendet oder empfangen werden, aus sieben oder acht Zeichen. In der Regel arbeiten Home-Computer mit acht bit-ASCII, da auch Grafikzeichen übertragen werden. Sieben bit ASCII erlaubt nur die Darstellung von Ziffern und Klein- und Großbuchstaben. Dem Zeichen, das übertragen wird, geht ein Start-bit voraus, damit der Empfänger weiß, daß nun ein Zeichen übertragen wird. Nach dem Zeichen kommen, je nach Protokoll, ein bit, das Paritäts-bit heißt und mit dessen Hilfe überprüft wird, ob das Zeichen richtig übertragen wurde. Abgeschlossen wird die Übertragung ebenfalls je nach Protokoll durch ein oder zwei Stop-bit.

Damit ist also ein Zeichen, das über die serielle Schnittstelle übertragen wurde, mindestens zehn bit lang. Genormt ist auch die Geschwindigkeit, mit der die Zeichen übertragen werden. Weit verbreitet ist die Geschwindigkeit von 300 oder 1200 Baud. Baud steht für bit je Sekunde. Schnelle Übertragungskanäle arbeiten mit 9600 Baud. Gängige Übertragungsgeschwindigkeiten sind 50, 75, 110, 150, 300, 600, 1200, 2400, 3600, 4800 und 9600 Baud.

Modems und Akustikkoppler arbeiten mit 300 oder 1200 Baud, während Drucker auch mit 9600 Baud „fertig werden". Diese Zahlen für die Übertragungsgeschwindigkeit klingen sehr hoch und lassen große Übertragungsleistungen vermuten. Die Probe zeigt jedoch, daß die serielle Schnittstelle nicht sehr schnell ist. Geht man davon aus, daß ein Zeichen mit Start- und Stop-bit zehn bit lang ist, so werden bei 300 Baud 30 Zeichen und bei 9600 Baud 960 Zeichen je Sekunde übertragen. Bei 9600 Baud ist dies zwar fast 1 KB, jedoch immer noch wenig, wenn man bedenkt, daß bei Personal-Computern der Datentransfer zwischen dem Hauptspeicher und einer Winchester-Platte mit fünf Megabit je Sekunde stattfindet, was etwa 500 KByte je Sekunde entspricht.

Für den, der die serielle Schnittstelle zur Kopplung von Computer und Peripherie einsetzt, ist es wichtig zu wissen, daß man eine solche Schnittstelle schrittweise aufbauen kann. Die einfachste Form ist die Zwei-Leitungs-Schnittstelle, bei der Daten vom Rechner zur Peripherie geschickt werden. Eine weitere Leitung sorgt für gemeinsames Potential. Erweitert wird diese einfache Schnittstelle um einen Datenkanal vom Peripheriegerät zum Computer. Die nächste Ausbaustufe besteht in der Verbindung der Leitung „Clear to send" und „Data Terminal ready". Dadurch wird sichergestellt, daß die Datenübertragung unterbrochen wird, wenn beim Empfänger der Buffer voll ist.

Bei der seriellen Schnittstelle, die auf einen 25poligen Stecker geführt ist, sind in der Regel nur die Pins 1 bis 7 und Pin 20 von Bedeutung. Mit diesen Signalen läßt sich fast immer eine voll funktionsfähige, bidirektionale serielle Datenübertragung aufbauen.

Daß serielle Schnittstelle nicht unbedingt gleich serielle Schnittstelle sein muß, zeigt die serielle Schnittstelle am Commodore VC 20 und C 64. Diese Schnittstelle arbeitet nicht mit den R 232 C-Standard-Signalen, sondern mit TTL (Transistor-Transistor-Logik-)Signalen. Während bei der RS 232 C-Schnittstelle eine logische Eins einer Spannung von mindestens −3 Volt und eine logische Null einer Spannung von mehr als +3 Volt entspricht, ist beim VC-20 eine logische Null eine Spannung zwischen 0 und 0,8 Volt und eine logische Eins eine Spannung größer als 2,4 Volt. Darin liegt der Grund, daß der VC-20 nicht an Geräte mit serieller Schnittstelle nach dem RS 232 C-Standard angeschlossen werden kann. Zur Spannungsanpassung ist eine Zusatzschaltung notwendig.

Scan: S. Höltgen

Stichwort: Schnittstellen (1)

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

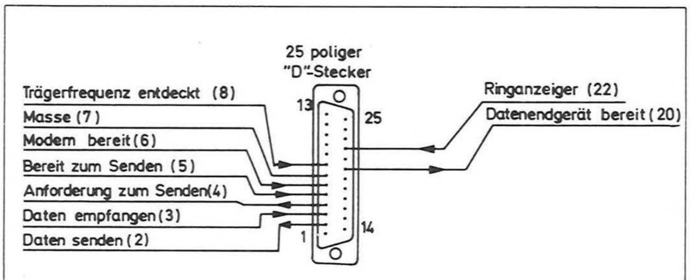
Serielle Schnittstelle

Datenübertragung zwischen verschiedenen Computer-Systemen oder zwischen Computer und Peripherie-Geräten, wie zum Beispiel Druckern und Plottern, erfolgt über Schnittstellen. In der Computer-Technik haben sich heute einige Standardschnittstellen einen festen Platz gesichert. Weit verbreitet sind die parallele und die serielle Schnittstelle. Zwei typische Vertreter der parallelen Schnittstelle sind die Centronics-Schnittstelle und der IEC-Bus. Beiden ist gemeinsam, daß Daten parallel übertragen werden. Das bedeutet, daß alle acht bit, die ein Zeichen bestimmen, wenn man den ASCII-Code benutzt (American Standard for Information Interchange), gleichzeitig an der Schnittstelle anliegen. Diese Schnittstelle besitzt daher acht Datenleitungen. Dazu kommen dann noch einige Übertragungsleitungen, mit denen der Datenfluß überwacht wird.

Serielle Schnittstellen übertragen die acht bit, die ein Zeichen ausmachen, hintereinander. Für die Datenübertragung im engeren Sinn ist nur eine Leitung notwendig, auf der die Bytes bitweise übertragen werden. Typische Vertreter dieser Schnittstelle sind die RS 232 C-Schnittstelle oder die Current-Loop-, auch 20-mA-Schleife genannt. Leider verfügt dieser Schnittstellentyp über keine einheitliche Normung. Gemeint ist damit, daß man in der Regel leider nicht mit einem Standardkabel zwei serielle Schnittstellen miteinander verbinden kann.

Trotzdem ist auch bei dieser Schnittstelle vieles durch Normen vereinbart. Ein Signal, dessen Spannung zwischen 5 und 25 Volt liegt, stellt eine logische Null, ein Signal zwischen −5 und −25 Volt stellt eine logische Eins dar.

Genormt ist auch die Art und Weise, in der Daten übertragen



Belegung einer seriellen Schnittstelle am Rechner

Scan: S. Höltgen

Stichwort: Schnittstellen (2)

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

Parallele Schnittstelle

Eine der einfachsten Schnittstellen, die die Computer-Technik kennt, ist die parallele Schnittstelle. Der Grund hierfür ist, daß auch auf dem internen Bussystem des Computers Daten parallel übertragen werden.

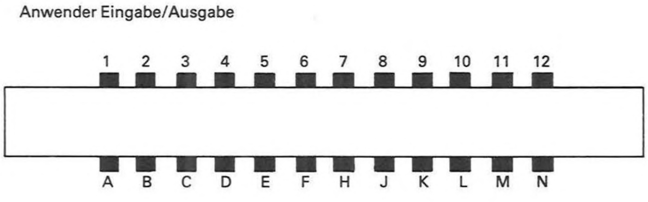
Nach einer internationalen Norm und bei 8-Bit-Prozessoren ist der Datenbus acht Bit breit, also genauso breit wie ein Byte. Dieser interne Kommunikationskanal entspricht also genau der Konvention, in der Daten gespeichert und verarbeitet werden.

Für die Kommunikation über eine parallele Schnittstelle sind neben den acht Datenleitungen nur noch zwei weitere Leitungen notwendig, mit denen der Datenaustausch zwischen Peripheriegeräten, also zum Beispiel einem Drucker, kontrolliert wird. Eine Leitung, die anzeigt, daß eine Informationseinheit zur Übertragung bereitsteht und eine andere, auf der das externe Gerät mitteilen kann, daß es bereit ist, Daten zu empfangen.

Centronics-Schnittstelle

Bei der Centronics-Schnittstelle, die im allgemeinen eingesetzt wird, um Drucker anzuschließen, gibt es noch weitere Verbindungsleitungen. Eine sorgt dafür, daß Sender und Empfänger im gleichen Takt arbeiten (Strobe), und Leitungen, auf denen dem Computer mitgeteilt wird, daß das Papier zu Ende ist, eine Leitung, die einen Fehler im Drucker mitteilt, und eine Verbindung, die den Drucker in den betriebsbereiten Zustand schaltet.

Da die Daten parallel, also alle Informationen, die zu einem Bit gehören, gleichzeitig, und nicht seriell, also hintereinander übertragen werden, ist die parallele Datenübertragung recht leistungsfähig. Zusammen mit vielen Druckern wird eine Übertragungsgeschwindigkeit von 1 KB je Sekunde erreicht, eine serielle



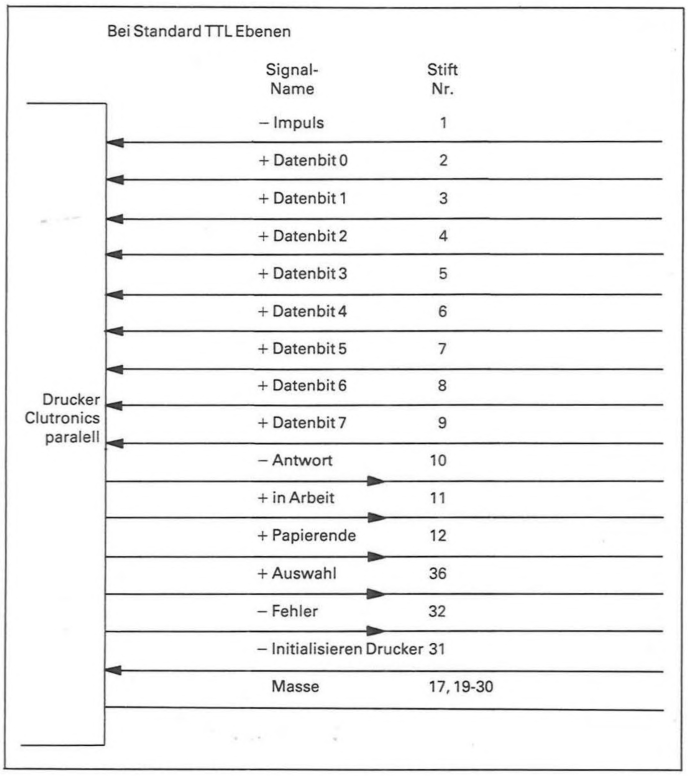
Pinbelegung für VC 20 auf der Rückseite

Scan: S. Höltgen

Schnittstelle müßte mit den entsprechenden Protokollbits mit 10 000 Baud arbeiten.

Bei vielen Computer-Systemen ist der parallele Datenbus an einem Stecker herausgeführt. Dieser Datenbus kann in der Regel nicht nur Daten nach außen, also vom Computer zur Peripherie übertragen, sondern auch umgekehrt. Dieser Anschluß eignet sich auch, um Daten von außen zu empfangen, also, um den Zustand von Schaltern zu überprüfen.

Eine Variante der parallelen Schnittstelle ist der User-Port an Commodore-Rechnern. Am User-Port sind die acht Datenleitungen herausgeführt. Mit POKE-Befehlen werden diese acht Leitungen als Ein- und Ausgabekanäle geschaltet. Ebenfalls mit POKE-Befehlen können Daten über diesen Anschluß ausgegeben und über PEEK-Befehle eingelesen werden. Allerdings folgen die Operationen über diese Schnittstelle den Gesetzen der Binärarithmetik. Die Werte, die an diesen Leitungen anliegen, entsprechen den Zahlen 2, 4, 8, 16, 32, 64 und 128. Eine Abfrage der Ein-/Ausgabekanäle ergibt diese Werte, wenn ein Signal anliegt. Mit diesen Zahlenwerten können die Kanäle auch auf die entsprechenden Werte geschaltet werden. Eine entsprechende Kombination ergibt die Zwischenwerte.



PIN #	TYPE	BEMERKUNG	PIN #	TYPE
1	GND		A	GND
2	+ 5 V	1ØØ mA MAX.	B	CB1
3	RESET		C	PBØ
4	JOYØ		D	PB1
5	JOY1		E	PB2
6	JOY2		F	PB3
7	LIGHT PEN		H	PB4
8	CASSETTE SWITCH		J	PB5
9	SERIAL ATN IN		K	PB6
1Ø	+ 9 V	1ØØ mA MAX.	L	PB7
11	GND		M	CB2
12	GND		N	GND

Scan: S. Höltgen

cht
on-
ce,
au-
ch-
us-
en
nen
efe-
alle
3 je-
ha-
ge-
uch
wir
ein-
lten!
ugs-
frei
Min-
sfrist
ende

Stichwort: Schnittstellen (3)

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

IEEE 488 oder IEC-Bus

Eine besonders leistungsfähige Form der parallelen Schnittstelle ist der IEEE-488-Bus. Diese parallele Schnittstelle wurde international genormt und findet sich in den Computern verschiedener Hersteller. Bei Commodore-Rechnern wurde sie als IEC-Bus implementiert, als HP-IB von Hewlett-Packard zur Verbindung der unterschiedlichsten Geräte mit Computer-Systemen konzipiert. Der besondere Vorteil dieser Schnittstelle liegt darin, daß sehr viele Geräte, maximal 16, miteinander verbunden werden können. Das Computer-System übernimmt als Controller die Verwaltung. Die anderen Geräte arbeiten als Empfänger (Listener) oder Sender (Talker). Von den Protokollvereinbarungen her kann nur ein Gerät Kontrollfunktion übernehmen und ein Gerät als Sender arbeiten, aber alle anderen Geräte als Empfänger.

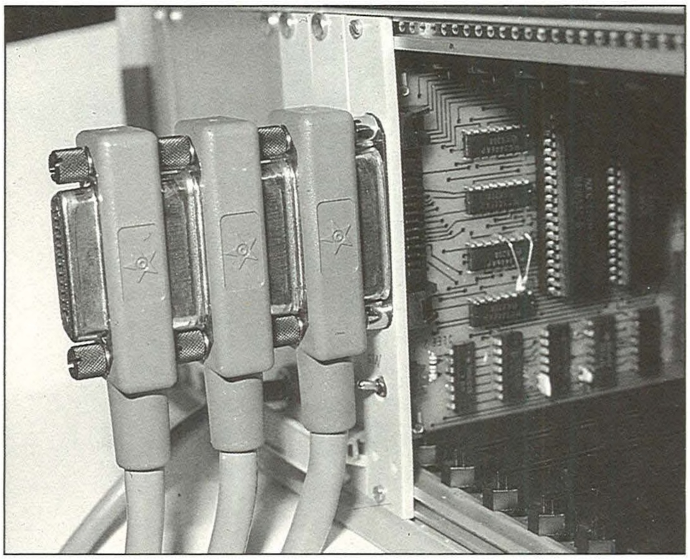
Neben den acht Datenleitungen gibt es Steuerleitungen für:

ATN (Attention) – vorher festgelegte Geräte können Daten übertragen.

DAV (Data valid) – die Daten sind gültig und können empfangen werden.

EOI (End of Identify) die Datenübertragung ist beendet.

IFC (Interface clear) – alle an den Bus angeschlossenen Ge-



IEC-Bus: Bis zu 16 Geräte sind anschließbar

Scan: S. Höltgen

räte werden in den Ausgangszustand gesetzt.

NDAC (No Data accepted) – diese Leitung signalisiert dem Sender, daß die Daten gelesen wurden.

NRFD (Not ready for data) – diese Leitung signalisiert, daß keine Daten empfangen werden können.

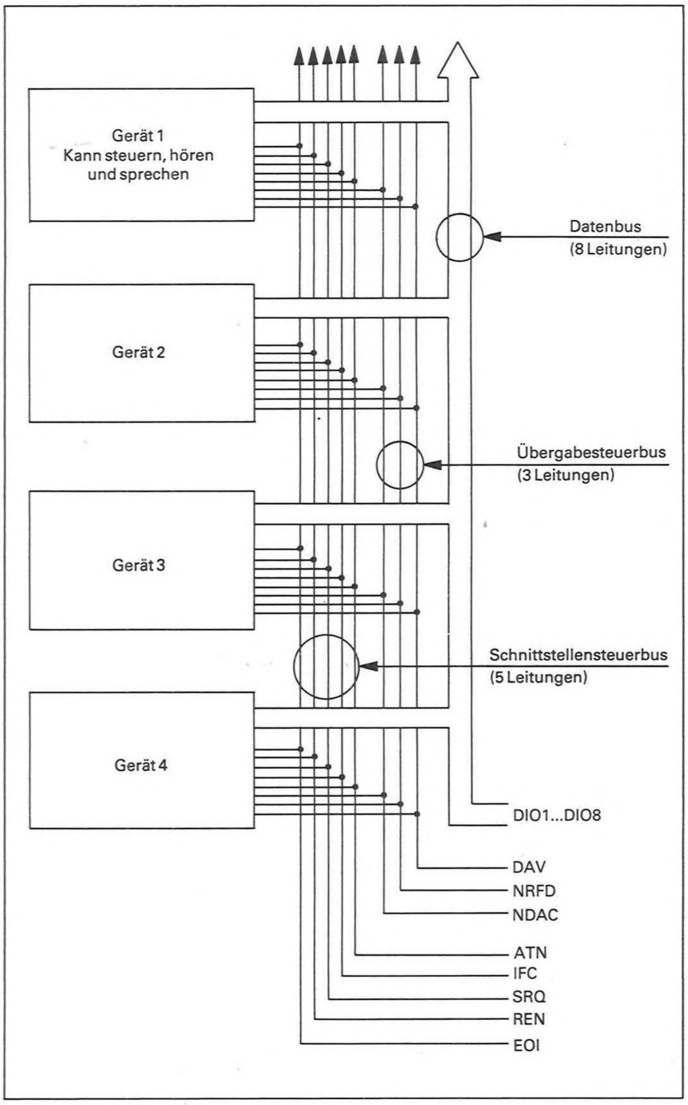
SRQ (Service requested) – ein peripheres Gerät kann die Übertragung von Daten anfordern.

REN (Remote enable) – mit diesem Signal werden Peripheriegeräte so geschaltet, daß sie nur noch vom IEEE-Bus aus gesteuert werden können.

Mit diesem Buskonzept können Computer, Empfänger wie Drucker, Sender wie Meßgeräte und Sender und Empfänger wie Diskettenstationen oder Grafiktabletts über eine einzige Schnittstelle miteinander verbunden werden. Welches Gerät welche Funktion ausführt, kann programmgesteuert festgelegt werden. Fest eingestellt an den angeschlossenen Geräten ist lediglich ihre Adresse. Eine Standardeinstellung ist:

0 für den Computer
4 für den Drucker
8 für die Diskettenstation

Beim HP-IB können auch mehrere Computer in einem Bussystem zusammenarbeiten, wobei jedoch nur einer als Controller arbeiten kann. Die anderen Systeme müssen durch entsprechende Programme als Controller abgeschaltet werden.



Aufbau des IEC-Bus

Scan: S. Höltgen

Stichwort: Schnittstellen (4)

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

HP-Interface-Loop

Um mehrere Geräte an einen Mikrocomputer anzuschließen, gibt es zwei Möglichkeiten: Entweder werden so viele Schnittstellen geschaffen, wie Geräte vorhanden sind, oder es wird eine Schnittstelle eingesetzt, an die mehrere Peripheriegeräte angeschlossen werden können.

Vorwiegend für den Laboreinsatz wurde der IEEE 488 entwickelt, an den bis zu 15 Geräte angeschlossen werden können. Für Sonderaufgaben wurde von Hewlett-Packard die HP-Interface-Loop entwikkelt, die nach ähnlichen Prinzipien arbeitet wie der IEEE 488, aber die Informationen nicht parallel, sondern seriell überträgt.

Schnittstellenschleife

Während der IEEE-488-Bus eine Schnittstelle ist, die überwiegend im Labor eingesetzt wird und relativ viel Strom verbraucht, ist die HP-IL eine bitserielle Schnittstelle. In ihrer Funktion mit dem IEEE-488-Bus identisch, kommt die HP-IL mit sehr geringen Spannungen und Strömen aus und benötigt lediglich zwei Kabel. Diese Schnittstelle wird von Hewlett-Packard bei den Taschenrechnern und portablen Computern eingesetzt, um Rechner, Kassettenlaufwerke, Drucker und Meßgeräte miteinander zu verbinden.

```
JEDE INFORMATION BESTEHT AUS 11 BIT

BIT 1    :  SYNC BIT
BIT 2,3  :  CONTROL BITS
BIT 4-11 :  DATA BITS

BEISPIELE:
"UNLISTEN"             100 00111111
"INTERFACE CLEAR"      100 10010000
"SEND DATA"            101 01100000
```

Datenstruktur des HP-IL

Diese einfache Schnittstelle überträgt die Information mit einem Elf-Bit-Code bei einer Übertragungsgeschwindigkeit von drei bis fünf Kilobyte je Sekunde. Übertragen auf die serielle Schnittstelle RS 232, bedeutet dies ungefähr 3000 bis 5000 Baud, was immerhin ausreicht, um in einer Sekunde eine DIN-A4-Seite Information zu übertragen.

Die HP-IL-Schnittstelle arbeitet mit ähnlichen Protokollen wie der IEC-Bus und erlaubt den Anschluß von 31 Geräten. Bei Erweiterung des Adreß-Codes kommt man auf maximal 961 Geräte. Die Signale haben einen Pegel von −1,5, 0 und +1,5 Volt.

Die HP-IL-Schnittstellenschleife ist ein geschlossener Stromkreis, in dem die einzelnen Peripheriegeräte zusammengeschlossen sind. Die ersten drei Bit der elf Bit langen Informationseinheiten, die im Kreis zirkulieren, enthalten

Scan: S. Höltgen

Steuerinformationen, mit deren Hilfe die Peripheriegeräte erkennen können, ob die Informationen für sie bestimmt sind. Jedes Byte, das in die Nachrichtenschleife geschickt wird, enthält diese Information.

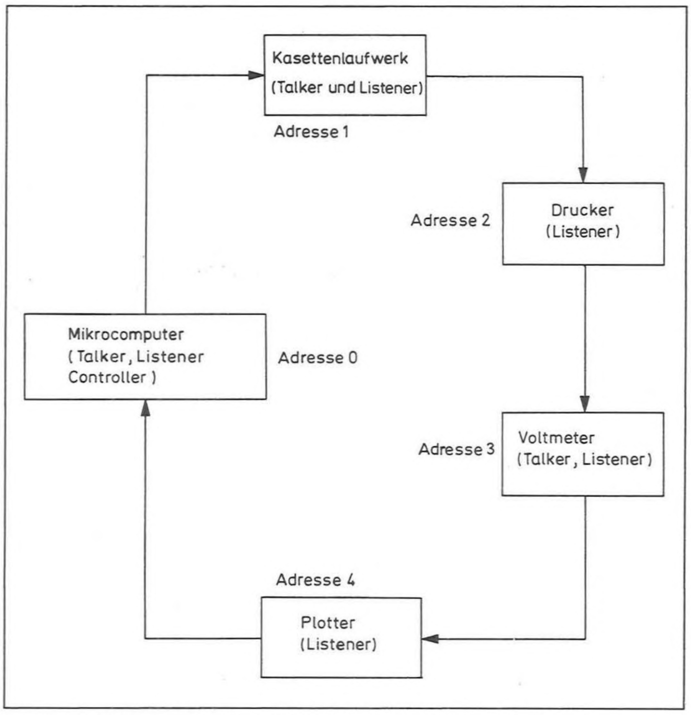
Für die Kommunikation mit dem HP-IL-Interface gelten einige Besonderheiten. Jedes Gerät, das in der Schleife angeschlossen ist, ist zunächst einmal Sender und Empfänger. Da die Nachrichten in der geschlossenen Schleife zirkulieren, muß jedes Gerät feststellen können, ob eine Nachricht an dieses Gerät adressiert ist. Schließlich gibt es eine ganze Reihe von Kontrollfunktionen, die die Nachrichtenübermittlung steuern. Darüber hinaus muß jedes Gerät als Verstärker arbeiten, denn die Information im Ring wird in jedem angeschlossenen Gerät verstärkt und weitergeschickt. Dadurch wird es möglich, auch mit den geringen Spannungen große Entfernungen zu überbrücken; mit einem Standardkabel kann die Entfernung zwischen zwei Geräten 10 Meter, mit einem abgeschirmten Kabel bis zu 1000 Meter betragen.

Das Nachrichtensystem verfügt über Steuerfunktionen, mit denen einzelne Geräte als Controller, Talker oder Listener geschaltet werden können. Von diesem Konzept her ist die HP-Interface-Loop sehr nahe mit dem IEEE 488 verwandt. Dazu hat sie aber auch einen Mechanismus, mit dem die Geräte, die zusammengeschlossen sind, abgeschaltet werden können.

Die Nachrichtenschleife wird in einen Schlafzustand geschickt, aus dem sie nur durch einen Controller oder einen Talker geweckt werden kann. Dadurch wird Strom gespart, wenn keine Daten zu übertragen sind.

Schnittstelle zu IEEE-488

Für die HP-IL gibt es inzwischen auch Schnittstellenmodule, an die IEEE-488-, serielle und parallele Schnittstellen angeschlossen werden können. Dadurch können Daten zwischen verschiedenen Computer-Systemen ausgetauscht oder an HP-IL-Geräten, zum Beispiel einem Taschenrechner, auch „normale" Peripheriegeräte, wie Drucker, Plotter oder Diskettenstationen, angeschlossen werden.



Ein Beispiel zum Aufbau einer Schnittstellenschleife

Scan: S. Höltgen

Stichwort: BASIC (1)

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

1. Darstellung von Zahlen

BASIC-Zeichen	Benutzungsweise	Bedeutung
	57.01 −1.2 .2549	Der Punkt ersetzt das Dezimalkomma. Ein Komma darf nicht geschrieben werden.
E	1.4 E 2 −1.4 E 3 1.4 E −4 Auch möglich: 2 E + 03 .004 E 4	Wissenschaftliche Schreibweise (E steht für Exponent = Hochzahl zu 10). 1.4 E 2 = 1.4 * 100 = 140 −1.4 E 3 = −1.4 * 1000 = −1400 $1.4\ E-4 = 1.4 * \frac{1}{10000} = 0.00014$ 2 E + 03 = 2000 .004 E 4 = 40 Der Exponent muß in gewissen Grenzen gewählt werden, die vom Gerät abhängen.

2. Arithmetische Befehle

BASIC-Zeichen	Benutzungsweise	Bedeutung
+ −	4 + 18 3 − 2.1	Addition, Subtraktion. Das Zeichen − dient auch als Vorzeichen.
* /	3 * 4.05 5/3 X / 2.5 * B	Multiplikation, Division. Der Doppelpunkt ist als Divisionszeichen nicht gestattet! Das Multiplikationszeichen darf nicht wegfallen, zum Beispiel muß die Formel 5 × + 9 als 5 * X + 9 geschrieben werden.
∧ ↑ **	2 ↑ 10 2 ↑ .5	Potenzieren einer beliebigen positiven Basis mit einer beliebigen Hochzahl. Beispiel: 2 ↑ .5 ergibt $\sqrt{2}$.
	5 ↑ (1/3) für $\sqrt[3]{5}$ 16/4/2 für (16/4)/2 das heißt 4/2 3 ↑ 3 ↑ 2 für 27 ↑ 2 und nicht 3 ↑ 9 3 ↑ (3 ↑ 2) für 3 ↑ 9	Arithmetische Ausdrücke werden von links nach rechts abgearbeitet, sofern nicht a) die üblichen Rechenregeln b) Klammern etwas anderes vorschreiben.

Scan: S. Höltgen

( )	3 * (4 + A)	Klammern verlangen, daß ihr Inhalt ausgerechnet wird, bevor andere Operationen ausgeführt werden. (Auch vor Klammern darf das Zeichen * nicht wegfallen.)

3. Eingebaute Funktionen

BASIC-Zeichen	Benutzungsweise	Bedeutung
SQR	SQR (3.2) SQR (3 * A + 1)	Quadratwurzel (engl. square root) einer nicht negativen Zahl
EXP	EXP (−14) EXP (.2*A + 1)	Exponentialfunktion e^x für beliebige Werte x; e = 2,71828... („Wachstumsfunktion")
LOG	LOG (3) LOG (2 * A) LOG (Ø.1)	Natürlicher Logarithmus (das heißt zur Basis e) einer positiven Zahl. (Manche Rechner, zum Beispiel PC 1211 und PC 1500, verwenden das Zeichen LN. LOG bedeutet dann den dekadischen Logarithmus.)
INT	INT (3.9) INT (−2.4) INT (5 * A)	Ergibt die nächstniedrigere ganze Zahl (engl. integer). Ganzzahlige Argumente werden nicht geändert. INT (3.9) ergibt 3. INT (−2.4) ergibt −3. INT (11) ergibt 11.
ABS	ABS (2) ABS (−4) ABS (A+1)	Ergibt den Absolutbetrag, das heißt ein negatives Vorzeichen fällt weg. In der Mathematik wird das Zeichen | | gebraucht.
SGN	SGN (7) SGN (−3) SGN (Ø)	Ergibt für das Vorzeichen (Signum) einer Zahl: +1 für positive Werte, −1 für negative Werte, Ø für Ø.
SIN COS	SIN (Ø.3) COS (Ø.4)	Sinus eines Winkels, der im Bogenmaß angegeben wird. Cosinus eines Winkels, der im Bogenmaß angegeben wird.
TAN ATN	TAN (Ø.5) ATN (2 *A)	Tangens eines Winkels, der im Bogenmaß angegeben wird. Arcustangensfunktion. Ergibt Winkel im Bogenmaß mit Werten zwischen $-\frac{\pi}{2}$ und $\frac{\pi}{2}$.
RND	RND (1)	Nimmt bei jedem neuen Aufruf einen zufälligen Wert zwischen Ø und 1 an. Über Einzelheiten zur Benutzung dieser Funktion und eventuelle Abweichungen gibt das Benutzerhandbuch Auskunft.
RANDO-MIZE	5Ø RANDOMIZE	Dieser Programmbefehl bewirkt, daß bei Wiederholung des Programmablaufs mit einer neuen Folge von Zufallszahlen gearbeitet wird.
	Zulässig: INT(1ØØ *LOG (ABS(X)))	Allgemein gilt: Funktionen können geschachtelt und mit arithmetischen Operationen zu längeren Formeln zusammengesetzt werden.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (2)

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

3. Eingebaute Funktionen

BASIC-Zeichen	Benutzungsweise	Bedeutung
	Nützliche Formeln:	
	4 * ATN(1) EXP (1) INT (A + .5)	Kreiszahl π Zahl e Runden der Zahl A auf die nächstliegende ganze Zahl.
	X – INT (X) ABS(X) – INT (ABS(X)) A – B * INT (A / B) PI = 4 * ATN (1) SIN (PI * A / 180) INT (6 * RND (1)) + 1 INT (49) * RND(1)) + 1)	Nachkommastellen für positives X. Nachkommastellen einer beliebigen Zahl X Rest von A bei Division durch B (A und B mit gleichen Vorzeichen) Ergibt den Sinus des Winkels A, wenn dieser im Gradmaß angegeben wird Eine zufällige „gewürfelte" Zahl von 1 bis 6 Eine Lottozahl

4. Variable

BASIC-Zeichen	Benutzungsweise	Bedeutung
LET... = ...	10 LET A = 2.718 20 LET B = A + 1 30 LET A = A + 1 40 LET A = SIN(10*B) 50 LET A = A / (A+1)	Hier steht links vom =-Zeichen eine Variable, das heißt der Name eines Speicherplatzes. Rechts steht ein Rechenausdruck, der zusammengesetzt sein kann aus Zahlen, Funktionen und Variablen. Diese Variablen sollten schon früher im Programm aufgetreten sein. Der Befehl veranlaßt die Berechnung des Ausdrucks unter Verwendung der aktuellen Werte der Variablen und weist das Ergebnis der links stehenden Variablen zu. Rechts und links darf dieselbe Variable auftreten wie etwa in Zeile 30, wo A um 1 erhöht wird. Also bedeutet = keine Gleichheit, sondern eine Wertzuweisung. Der Sachverhalt wäre besser beschrieben durch einen Pfeil, zum Beispiel A ← B+1.

Scan: S. Höltgen

BASIC-Zeichen	Benutzungsweise	Bedeutung
	Auch möglich: 10 A = 2.718 20 B = A + 1	Die Reihenfolge, in der Wertzuweisungen vorgenommen werden, ist wesentlich. Das Wort LET ist nicht unbedingt erforderlich, es wird aus Bequemlichkeit meist weggelassen.
	Falsch: 70 A = B = 5 Richtig: 70 A = 5 : B = 5	Mehrfache Wertzuweisungen durch eine Kette von Gleichheitszeichen sind nicht möglich.
	Nützlich: 100 LET A = A + B 110 LET B = A − B 120 LET A = A − B	Vertauschung der Werte, die in A und B gespeichert sind, ohne Benutzung einer dritten Variablen.
%	150 LET B% = 120	Wenn von vornherein feststeht, daß eine Variable nur ganzzahlige Werte besitzen soll, ist es aus maschineninternen Gründen zweckmäßig, an den Variablennamen % anzuhängen. Nichtganze Werte werden dann abgerundet. (Diese Möglichkeit ist nicht bei allen Rechnern vorgesehen; auch ist die Symbolik nicht einheitlich.)
	A, B, C A0, H2, O9 RA, ZW, EB Meist zulässig: WERT MARTIN SIGNAL MAERZ Nicht zulässig: WIRT-SCHAFTS-INDEX MONAT INTEGRAL STANDARD FAKTOR	Als Namen von Variablen kommen in Frage: a) Einzelbuchstaben, b) ein Buchstabe und eine Ziffer, häufig auch: c) Buchstabenpaare. Viele Computer erlauben auch längere Variablennamen, jedoch werden häufig nur die ersten beiden Buchstaben beachtet (zum Beispiel bei APPLE II, VC 20). Andere Geräte (zum Beispiel ZX 81) beachten alle Zeichen. Vorteil längerer Namen: bessere Lesbarkeit und Verständlichkeit des Programms. Nachteil: Fehlerquelle bei unbeabsichtigter Verwendung zweier Namen mit übereinstimmenden ersten und zweiten Buchstaben (zum Beispiel MARTIN und MAERZ). Nicht zulässig sind Variablennamen, die eine für BASIC-Befehle oder -Funktionen reservierte Zeichenfolge enthalten. Eine Liste der reservierten Wörter finden Sie in dem Handbuch Ihres Computers.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (3)

Die wichtigsten Begriffe aus der Computertechnik – in Stichworten zusammengefaßt

5. Texte

BASIC-Zeichen	Benutzungsweise	Bedeutung
$	10 LET A$ = "DM" 20 LET B$ = "*****"	1) Kennzeichnet den Speicherplatz eines Textstücks (sogenannter STRING), der Buchstaben, Ziffern und Symbole enthalten darf. 2) Kennzeichnet Textfunktionen.
+	30 LET C$ = B$ + A$	Hängt Zeichenketten aneinander. Im Beispiel entsteht *****DM.
LEN(X$)	50 PRINT LEN(" DM") 60 LET A = LEN(A$+B$)	Gibt die Länge einer Zeichenkette an. Ergebnis von Zeile 50: 3. Zeile 60 weist A den Wert 7 zu.
LEFT$ (A$,N)	70 PRINT LEFT$ (C$,4) 80 LET D$= LEFT$ (A$,2*I)	Gibt die ersten N-Zeichen einer Kette wieder. Zeile 70 ergibt aufgrund der Zeilen 10 bis 30: ****. (N=0 ist nicht erlaubt.)
RIGHT$ (A$,N)	wie LEFT$	Gibt die letzten N-Zeichen einer Kette wieder.
MID$ (A$,M,N)	wie LEFT$	Gibt N-Zeichen einer Kette ab der M. Stelle wieder. N darf den Wert 0 haben; das Ergebnis ist dann der sogenannte Leerstring "".
STR$ (X)	100 LET X = 22/7 110 LET X$ = STR$ (X) 120 PRINT LEFT$ (X$,4)	Verwandelt den in X gespeicherten Zahlenwert in eine Zeichenkette. Im Beispiel wird 3.14 ausgedruckt.
VAL (A$)	200 PRINT VAL ("3,50DM") 210 PRINT VAL ("3,50")	Wandelt eine Zeichenkette, die mit Ziffern beginnt, in die entsprechende Zahl um. Beginnt die Textkette nicht mit einer Ziffer, entsteht die Zahl 0. Durch die Zeile 200 erscheint 3.5, durch Zeile 210 erscheint 3.
ASC (X$)		Für jedes Zeichen ist ein Zahlenwert zwischen 0 und 255 vereinbart worden (ASCII-Code). Der Befehl gibt diesen Wert für das erste Zeichen der Kette wieder.
CHR$ (X)		Gibt das zur Zahl X gehörige Zeichen an.

Scan: S. Höltgen

6. Anzeige

BASIC-Zeichen	Benutzungsweise	Bedeutung
PRINT...	200 PRINT E 210 PRINT A$ 220 PRINT Z% 230 PRINT 2*A*SIN(T)	Der aktuelle Wert der Variablen nach dem PRINT-Befehl (englisch drukken) wird ausgedruckt beziehungsweise auf dem Bildschirm geschrieben. Auch zusammengesetzte Ausdrücke dürfen auftreten.
PRINT ".."	250 PRINT "AUSWERTUNG"	Der in Anführungszeichen stehende Text wird gedruckt.
PRINT...;	300 PRINT "LOESUNG X="; X	Durch ; getrennte Ausdrücke werden ohne Zwischenraum hintereinander geschrieben. Das Resultat von Zeile 300 könnte sein: LOESUNG X=.482692
	350 PRINT A; 360 PRINT "...ZU KLEIN!"	Das Semikolon am Ende eines PRINT-Befehls bewirkt, daß beim nächsten Druckbefehl nicht automatisch eine neue Zeile begonnen wird. Das Ergebnis der Zeilen 350 und 360 könnte sein: 2...ZU KLEIN!
PRINT...,	400 PRINT X1, X2, X3 410 PRINT I, SQR(I) 420 PRINT , A	Das Schreibfeld ist in vertikale Zonen eingeteilt. Der auf das Komma folgende Ausdruck wird in der nächsten Zone linksbündig geschrieben (Tabellenform). Mögliches Ergebnis der Zeilen 400, 410 und 420: 1.63 −0.4 2.916 625 25 7.0112
TAB (N)	500 PRINT TAB (10); A	Die Tabulator-Anweisung ist Teil eines PRINT-Befehls. Der Wert des folgenden Ausdrucks wird ab Spalte N geschrieben. Das Argument von TAB darf ein algebraischer Ausdruck sein, der auch Variable enthält; Dezimalwerte werden gerundet.
SPC	200 PRINT X; SPC(3); Y	Erzeugt eine bestimmte Anzahl von Leerfeldern zwischen aufeinanderfolgenden geschriebenen Werten.
PRINT	600 PRINT	Erzeugt eine Leerzeile.
	Nützlich: 100 A$ = "BELIEBIG" 110 L = LEN (A$) 120 PRINT TAB((40-L) /2); A$ 130 PRINT : PRINT	Schreibt den in A$ gespeicherten Text zentriert (hier über 40 Spalten). Programmzeile 130 erzeugt anschließend zwei Leerzeilen.
PRINT USING		Erlaubt das Ausdrucken in einem bestimmten Format (Maske). Nicht in allen BASIC-Dialekten vorhanden.
PRINT AT		Wirkung ähnlich wie PRINT TAB. Nicht in allen BASIC-Dialekten vorhanden.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (4)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

7. Sprungbefehle

BASIC-Zeichen	Benutzungsweise	Bedeutung
GO TO	50 GO TO 20 Auch möglich: 50 GOTO 20 Häufig nicht gestattet: 70 GO TO K	Erzwungener Sprung zu der betreffenden Programmzeile. Die angegebene Zeilennummer muß in dem betreffenden Programm tatsächlich auftreten. Die Zeilennummer ist eine feste Zahl, Variable sind meist nicht gestattet (Ausnahme z. B. ZX 81)
GO TO	Sinnlos: 100 GO TO 100	Ein Sprung zu der eigenen Programmzeile ist möglich aber i. a. nicht sinnvoll: Das Programm bleibt dort „hängen".
GO TO (Steuerbefehl)	GO TO 100	GO TO kann auch benutzt werden, um ein im Speicher geladenes Programm an einer bestimmten Stelle zu beginnen. Werte von Variablen werden dabei nicht gelöscht. (Vgl. RUN)
ON...GO TO	100 ON L GO TO 50, 90, 400, 999	Es erfolgt ein Sprung zu der 1., 2., 3., ... Zeilennummer, je nachdem, ob die Variable, hier L, den Wert 1, 2, 3, ... besitzt.
	120 ON B GO TO 10, 10, 20, 30, 30, 20	Dieselbe Zeilennummer darf mehrmals auftreten. Hat die Variable keinen passenden Wert, etwa weil sie zu klein, zu groß oder nicht ganzzahlig ist, so kann ein Abbruch des Programms erfolgen. Diese Fälle sind in den einzelnen BASIC-Dialekten unterschiedlich geregelt. Der ON...GO TO-Befehl ist besonders bei interaktiver Programmiertechnik interessant („Menü-Technik"). Er kann mehrere logische Abfragen ersetzen.
	Auch möglich: 100 ON A-3 GOTO ... 200 ON INT (X/6) GOTO..	Statt einer Variable darf auch ein komplexer Ausdruck verwendet werden. GOTO darf auch ohne Zwischenraum geschrieben werden.

Programmbeispiele:

10 PRINT "WURZEL-TABELLE": PRINT 20 I = 1	Dieses Programm berechnet die Wurzeln der Zahlen 1, 2, 3, ... Es „hängt" in einer endlosen Schleife

Scan: S. Höltgen

30 PRINT "WURZEL AUS";I;"=";SQR(I) 40 I = I + 1 50 GOTO 30 60 END	und kann nur durch äußeren Eingriff beendet werden (solange 1 zulässige Werte annimmt).
10 PRINT "WAEHLEN SIE: 1 – SCHNELLE METHODE" 20 PRINT TAB (13); "2 – EINF. GENAUIGKEIT" 30 PRINT TAB (13); "3 – DOPPELTE GENAUIGKEIT" 40 INPUT W 50 ON W GO TO 1000, 4000, 7000	„Menü"-Technik bei Dialogbetrieb

8. Schleifen

BASIC-Zeichen	Benutzungsweise	Bedeutung
FOR A = ...TO... NEXT	30 FOR A = 10 TO 23 40... 50 NEXT A	Die zwischen dem ersten (FOR) und dem letzten (NEXT) Befehl stehenden Anweisungen werden mehrfach durchlaufen. Bei Erreichen des Befehls NEXT wird ein neuer Durchlauf mit um 1 erhöhtem A-Wert begonnen. Das geschieht solange der aktuelle Wert von A den Endwert nicht übersteigt.
	30 FOR X = I TO 2 * K 40 ... 50 NEXT X	Anfangs- und Endwert können in den meisten BASIC-Dialekten durch algebraische Ausdrücke angegeben werden.
	30 FOR X = 1 TO 100 40 FOR Y = -1 TO 1 60 ... 80 NEXT Y 100 ... 140 NEXT X	Mehrere FOR ... NEXT-Schleifen – natürlich mit verschiedenen Laufindizes – können ineinander geschachtelt werden (i. a. bis zu 10 Schleifen).
	Auch möglich: 140 NEXT Y, X	Fallen die Enden mehrerer ineinander geschachtelter Schleifen zusammen, so können die NEXT-Befehle zusammengefaßt werden. Dabei ist auf die richtige Reihenfolge der Variablen zu achten.
	Nützlich: 1000 FOR W = 1 TO 1E4 1010 NEXT W	Warteschleife: Eine solche kontrollierte Verzögerung kann z. B. bei der Bildschirmausgabe von Nutzen sein.
	100 FOR I = 2 TO 1 140 ... 180 NEXT	Schleifen mit unsinnigen Begrenzungen werden bei einigen Computern einmal durchlaufen, bei anderen übersprungen.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (5)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

BASIC-Zeichen	Benutzungsweise	Bedeutung
STEP	30 FOR A=1 TO 5 STEP 2 50 … 60 NEXT A	Der Laufindex wird jeweils um den hinter STEP (engl. Schrittweite) angegebenen Wert erhöht. „STEP 1" darf entfallen.
	30 FOR X=1 TO 9 STEP 4 40 … 50 NEXT X	Beliebige Dezimalzahlen, auch negative, sind als Schrittweite zulässig.
	30 FOR A = 20 TO 10 STEP -3 50 … 60 NEXT A	Die Zählung muß nicht genau aufgehen. A nimmt hier die Werte 20, 17, 14, 11 an.
		Vorsichtsmaßnahmen: – Der Wert der Laufvariablen sollte beim Schleifendurchlauf nicht aktiv geändert werden. (Gleiches gilt für den Schleifen-Endwert.) – Programmsprünge in eine Schleife hinein sind zu vermeiden. – Mehrere Schleifen dürfen sich nicht kreuzen. – Das Programm gewinnt oft an Übersichtlichkeit, wenn verschiedene Schleifen auch verschiedene Laufindizes besitzen.

9. Dateneingabe

BASIC-Zeichen	Benutzungsweise	Bedeutung
INPUT	10 INPUT A 20 INPUT A$	Beim Antreffen des Befehls INPUT wird die Ausführung des Programms unterbrochen und die Eingabe des Wertes der folgenden Variable über die Tastatur abgewartet. Die Eingabe wird durch die RETURN-Taste beendet, und das Programm wird weiterbearbeitet.
	100 INPUT X, Y, Z Eingabe z. B.: 2.51, 5.936, 6.06	Es dürfen auch mehrere Werte gleichzeitig angefordert werden. Bei der Eingabe werden sie durch Kommata getrennt.

Scan: S. Höltgen

	110 INPUT P, W$ Eingabe z. B.: 567337, "AUGUST" oder auch: 567337, AUGUST	Bei der Eingabe von Texten (für Stringvariablen) kann häufig auf Anführungszeichen verzichtet werden. Sie sind jedoch erforderlich, wenn Leerzeichen oder Kommata aufgenommen werden sollen oder am Anfang Ziffern stehen.
	Beispiel: 10 PRINT "EINGABE: IHR MESS-WERT"; 20 INPUT M Häufig möglich: 10 INPUT "EINGABE: IHR MESS-WERT"; M	Es ist dringend zu empfehlen, INPUT-Befehle zu erläutern, etwa wie im Beispiel. Der vorangestellte PRINT-Befehl kann bei manchen Computern mit dem INPUT-Befehl kombiniert werden. Die Eingabevariable muß dann durch ein Semikolon vom Text getrennt werden.
GET (INKEY$) (KEY$)	100 GET A 110 GET Z$ bzw. 120 Z$ = INKEY$	(Nicht bei allen Computern vorhanden.) Der Computer fragt ab, ob und welche Taste gedrückt wurde. Manche Computer halten die Ausführung des Programms an, solange keine Taste betätigt wurde. Für praktische Zwecke ist die Benutzung einer Stringvariablen vorzuziehen, denn ein nicht-numerisches Zeichen führt bei Programmzeile 100 zu Fehlermeldung und Abbruch des Programms.
	30 GET A$ 40 IF A$= ""THEN GOTO 30 Nützlich: 100 PRINT "WEITER? (LEERTASTE DRUECKEN)" 110 GET A$ 120 IF ASC (A$) <>32 THEN GOTO 110	Einige Computer halten bei GET nicht an, sondern laufen auch ohne Tastendruck weiter. Der Variablen A$ wird in diesem Fall der Leerstring "" zugeordnet. Soll der Computer anhalten, so programmiert man eine Warteschleife (Zeilen 30 und 40, vgl. Kapitel 10). Die Ausführung des Programms wird unterbrochen, bis der Benutzer die Leertaste drückt. Andere Tasten bewirken nichts. (32 ist der ASCII-Wert der Leertaste.)
	Nützlich: 10 PRINT "IHRE WAHL:" 20 GET A$ 30 IF ASC (A$)<49 OR ASC(A$)>51 THEN GOTO 20 40 ON VAL (A$) GOTO ...	Schnelle Menü-Technik: Drücken der Zifferntasten 1, 2 oder 3 bewirkt den Sprung an die jeweils vorgesehene Adresse. Die RETURN-TASTE wird nicht benötigt.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (6)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

BASIC-Zeichen	Benutzungsweise	Bedeutung
READ DATA	50 READ A, B 60 READ C 1000 DATA .74, -1E3, 1	Den Variablen im READ-Befehl (engl. Lesen) werden die bei DATA festgeschriebenen Werte zugeordnet. Im Beispiel erhält A den Wert 0.74, und B erhält den Wert -1000. Es dürfen mehrere READ-Befehle im Programm auftreten. Die vorhandenen Daten werden der Reihe nach verarbeitet. C erhält hier den Wert 1.
	Beispiel: 10 FOR I = 1 TO 8 20 READ V(I) 30 NEXT I ... 1000 DATA 4,6,13,12,9 1001 DATA 2,7,10,0,1 Zulässig: 10 DATA "MEISTER", 1209 ... 1000 READ N$, W	Es dürfen mehr Daten bereitgestellt werden als tatsächlich gelesen werden. Es dürfen mehrere DATA-Befehle auftreten. Die Stellung der DATA-Befehle im Programm ist beliebig: Sie dürfen auch vor dem READ-Befehl stehen.
RE-STORE	10 READ X, Y ... 80 RESTORE 90 READ K, L, M, N ... 1000 DATA 1.6, -2, 3, 9	Der RESTORE-Befehl bewirkt das Einlesen der Daten wieder ab der ersten DATA-Anweisung: Die gleichen Daten können damit in mehreren Rechnungen verwandt werden. Im Beispiel erhalten sowohl X als auch K den Wert 1.6 und sowohl Y als auch L den Wert -2.
		Bei einigen Computern (z. B. ZX 81) existieren die Befehle READ, DATA nicht. Es gibt jedoch i. a. dazu gleichwertige Prozeduren.

10. Abfragen

BASIC-Zeichen	Benutzungsweise	Bedeutung
IF... THEN	100 IF A=0 THEN B=2*B	Zwischen IF und THEN steht eine mathematische Aussage. Es wird

Scan: S. Höltgen

BASIC-Zeichen	Benutzungsweise	Bedeutung
	200 IF LOG(X)>87.5 THEN PRINT "X ZU GROSS" 300 IF A=0 THEN PRINT "LOG(A) NICHT ERKLÄREN"	überprüft, ob diese Aussage wahr oder falsch ist. Ist sie wahr, so wird der Befehl hinter THEN ausgeführt. Ist sie falsch, so erfolgt sofort ein Sprung zur folgenden Zeilennummer.
	400 IF I <> INT(I) THEN PRINT "I NICHT GANZ"	Mathematische Aussagen ergeben sich in erster Linie aus Größenbeziehungen. Dazu werden zwei arithmetische Ausdrücke den Zeichen = (gleich) > (größer als) < (kleiner als) <> (ungleich) > = (größer oder gleich) < = (kleiner oder gleich) entsprechend verglichen.
	500 IF A$ = "HUBER" THEN ...	Stringvariable sollten nur auf Gleichheit oder Ungleichheit überprüft werden.
		Das „=“-Zeichen hat bei Aussagen seine aus der Mathematik bekannte Bedeutung. Es handelt sich nicht um eine Wertzuweisung.
AND OR NOT	500 IF A=0 AND B=12 THEN.. 600 IF X<0 OR (X>0 AND Y>0) THEN... Nützlich: 700 INPUT "J/N"; I$ 710 IF NOT (I$="J" OR I$="N") THEN GOTO 700 720 IF I$="J" THEN...	Mehrere Aussagen können auch zusammengesetzt werden. Dazu stehen zur Verfügung: AND (und) OR (oder) NOT (nicht). Es gelten die Regeln der Aussagenlogik; insbesondere ist OR nicht im ausschließenden Sinn zu verstehen. Programm läuft erst nach eindeutiger Antwort (JA/NEIN) durch den Benutzer weiter.
	Richtig: 50 IF A>10 THEN B=B+1: GOTO 100 Falsch: 50 IF A>10 THEN B=B+1 AND GOTO 100	Stehen nach einer Abfrage mehrere durch „:“ getrennte Befehle, so werden entweder alle (wenn die Aussage wahr ist) oder keiner (wenn die Aussage falsch ist) ausgeführt. AND, OR, NOT verknüpfen nur mathematische Aussagen, nicht Befehle.
	100 FOR I=1 TO N-1 110 FOR J=1 TO N-1 120 IF A(J)>A(J+1) THEN C=A(J):A(J)=A(J+1):A(J+1)=C 130 NEXT J 140 NEXT I	2. Sortieren der im Feld A(0)... A(N) gespeicherten Zahlen der Größe nach.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (7)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

BASIC-Zeichen	Benutzungsweise	Bedeutung
IF... THEN	Nützlich: 100 IF B-A< 1E-8 THEN GOTO 200 110 C= (A+B)/2 120 IF FN F(A)*FN F(C)<0 THEN B=C : GOTO 100 130 A=C 140 GOTO 100 200 PRINT "NULLSTELLE BEI "; A; "..."; B	1. Nullstellenbestimmung der Funktion FN F(X) durch Intervallhalbierung. Vorausgesetzt wird, daß das Anfangsintervall genau eine Nullstelle enthält.
	90 MA = -1E38 100 FOR I = 1 TO N 110 IF MA<A (I) THEN MA=A(I)	2. Suchen nach dem größten Wert des Feldes A(..).
	120 NEXT I 130 PRINT "MAXIMUM="; MA 10 A = SIN(X) 20 IF A=0.4 THEN... Besser: 20 IF ABS (A-0.4) <1E-8 THEN... 10 INPUT A 20 IF A↑2 = 9 THEN... Besser: 20 IF INT (A↑2 +.1) = 9 THEN...	3. Sicherung gegen Rundungsfehler: Alle berechneten Werte können Rundungsfehler aufweisen, und unter Umständen erkennt der Computer aus diesem Grund die Gleichheit von zwei Zahlen nicht. Deswegen ist die zweite Abfrage sinnvoller. Real-Arithmetik kann auch im Bereich der ganzen Zahlen zu Rundungsfehlern führen, so daß die Gleichheit von 3↑2 und 9 nicht erkannt wird. Im Bereich der ganzen Zahlen hilft die INT-Funktion.

Scan: S. Höltgen

11. Technik bedingter Sprünge

BASIC-Zeichen	Benutzungsweise	Bedeutung
IF... THEN (GOTO)..	100 IF A = 1000 THEN GOTO 10 Oft auch möglich: 100 IF A = 1000 THEN 10	In vielen Fällen folgt auf eine Abfrage ein Sprungbefehl. Man nennt dies einen bedingten Sprung. In vielen BASIC-Dialekten darf das Wort GOTO bei einem bedingten Sprung entfallen.
	Nützlich: 10 INPUT " GEBEN SIE EINE ZAHL ZWISCHEN 10 und 20 EIN" ; X 20 IF ABS(X-15)>5 THEN 10	Überprüfen von Eingaben: Werte für X, die außerhalb des gesetzten Intervalls liegen, werden zurückgewiesen. Das Programm hängt so lange an dieser Stelle, bis ein zulässiger Wert eingegeben wird.
	Beispielprogramm: 10 X=0 20 PRINT X, SIN(X), COS(X) 30 X=X+.01 40 IF X < 4*ATN(1) THEN 20 50 END	Offene Schleife: In Schritten von 0.01 wird eine Tabelle der SIN- und COS-Funktionen geschrieben. Das Programm wird beendet, wenn X größer als π wird (Alternative zu einer FOR...NEXT-Schleife).
		Oft ist die Anzahl der Wiederholungen einer Schleife nicht von vornherein bekannt; gerade sie kann die interessierende Größe sein.
	Nützlich: 10 INPUT K, T, P 20 IF K<=0 THEN 60 30 K = K * (1+P/1200)-T 40 N = N + 1 50 GOTO 20 60 M = INT(N/12) 70 PRINT" TILGUNGS-DAUER:" 80 PRINT M; "JAHRE"; N-12*M; "MONATE." 90 END	Das Programm berechnet die Tilgungsdauer eines Kredits von K DM bei P% jährlicher Verzinsung und einer monatlichen Tilgungsrate T.

12. Unterprogramme

BASIC-Zeichen	Benutzungsweise	Bedeutung
GOSUB RETURN	200 GOSUB 1000 210 PRINT A ... 1000 ... 1010 ... 1100 RETURN	Der Rechner springt zu der nach dem GOSUB-Befehl stehenden Zeilennummer und arbeitet den dort beginnenden Programmteil (das „Unterprogramm") ab. Erreicht er dabei den Befehl RETURN, so fährt er mit der Zeile nach dem GOSUB-Befehl fort. Deren Nummer ist vorübergehend als Rücksprungadresse intern notiert worden.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

		verwenden (z. B. AU, BU, ...). Deren Anfangswerte legt man im aufrufenden Programm vor dem GOSUB-Befehl fest.
Programmbeispiele		
10 INPUT N 20 INPUT K 30 XU = N 40 GOSUB 1000 50 B = FAK 60 XU = K 70 GOSUB 1000 80 B = B/FAK 90 XU = N − K 100 GOSUB 1000 110 B = B/FAK 120 PRINT B 130 STOP 1000 FAK = 1	1010 FOR I = 1 TO XU 1020 FAK = FAK * I 1030 NEXT I 1040 RETURN 2000 END	Berechnet den Binomial-Koeffizienten $\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$ In Zeile 1000 beginnt das Unterprogramm, das die Fakultät der Zahl XU berechnet. Es wird dreimal aufgerufen. Wichtig ist der Befehl STOP in Zeile 130. Entfiele er, so würde das Unterprogramm ein viertes Mal mit dem aktuellen Wert von XU ausgeführt, ohne aufgerufen worden zu sein. Bei Erreichen von Zeile 1040 erschiene dann eine Fehlermeldung.

13. Benutzerdefinierte Funktionen

BASIC-Zeichen	**Benutzungsweise**	**Bedeutung**
	10 DEF FN F(X) = .3*X*X+4*X+1 20 C=FN F(2) 30 E = .6*FN F(-A*B+INT(SQR(A)))	Zeile 10 gibt die Rechenvorschrift für eine Funktion FN F(X) an, die im Verlauf des Programms für verschiedene Werte von X berechnet werden soll. Im Beispiel wird der Wert 10.2 unter C gespeichert.
	DEF FN..(..) 40 DEF FN KI(X) = SQR(A*A−X*X) Ausnahmsweise möglich: 50 DEF FN G(X,Y)= LOG(X)+Y	Folgende Regeln sind zu beachten: 1. Als Funktionsnamen kommen dieselben Namen wie für Variable in Betracht. 2. Die definierende Zeile muß vor dem erstmaligen Aufruf im Programm stehen. 3. Für die Variable in der definierenden Zeile wird kein Speicherplatz reserviert (sog. „dummy variable“ = Blindvariable). Sie kann zusätzlich als gewöhnliche Variable verwendet werden. 4. Die definierende Rechenvorschrift darf auch gewöhnliche Variable enthalten. Bei Berechnung der Funktion werden dann die aktuellen Werte dieser Variablen benutzt. 5. Häufig lassen sich nur Funktionen **einer** Variablen durch eine DEF-Zeile definieren.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (8)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

12. Unterprogramme

BASIC-Zeichen	Benutzungsweise	Bedeutung
GOSUB RETURN	200 GOSUB 1000 210 PRINT A ... 1000... 1050 GOSUB 2000 ... 1100 RETURN 2000 ... 2010... 2100 RETURN	Innerhalb eines Unterprogramms kann in ein weiteres gesprungen werden. Die Rücksprungadressen werden der Reihe nach notiert und in umgekehrter Reihenfolge bei den einzelnen RETURN-Befehlen verwendet. („Last in – first out") Die Anzahl der möglichen Unterprogrammebenen ist bei den einzelnen Computern verschieden.
ON... GOSUB	300 ON T GOSUB 1000, 1500, 4000, 3000	Es erfolgt ein Sprung zur 1., 2., 3., ... Zeilennummer hinter GOSUB, je nachdem, ob T den Wert 1, 2, 3, ... besitzt.
	350 ON INT(.5*X-3) GOSUB...	An die Stelle der Variablen T kann auch ein zusammengesetzter algebraischer Rechenausdruck treten. Ist der Rechenausdruck nicht ganz oder kleiner als 1 oder größer als die Anzahl der gegebenen Sprungadressen, so ist das Verhalten der einzelnen Computertypen unterschiedlich. Meist wird ein nichtganzer Wert durch die INT-Funktion gerundet.
		Variable in Unterprogrammen: Prinzipiell dürfen in einem Unterprogramm dieselben Variablen wie im aufrufenden Programm auftreten.
		Beim Ablauf des Unterprogramms werden dann die aktuellen Werte der Variablen benutzt. Der Sinn von Unterprogrammen besteht jedoch darin, daß sie in den verschiedensten Programmteilen aufgerufen werden können und sogar Standardbestandteile verschiedener Programme sein können. In diesem Fall ist es zweckmäßig, im Unterprogramm eigene Variable zu

Scan: S. Höltgen

Stichwort: BASIC (9)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

14. Variablenfelder

BASIC-Zeichen	Benutzungsweise	Bedeutung
DIM..(..)	10 DIM A(100) 20 DIM B(5), C(6)	Reserviert Speicherplätze für eine Reihe von Variablen, die die Namen A(0), A(1), ... A(100) erhalten. Diese Variablen bilden ein sogenanntes Feld (auch Vektor genannt). Die Laufzahl, hier zwischen 0 und 100, des Feldes heißt Index (engl. „subscript"). Anstelle einer Zahl kann auch ein algebraischer Ausdruck eingesetzt werden. Voraussetzung ist, daß sein aktueller Wert > = 0 ist. Bei einigen Computern entfällt der Index 0.
		Alle durch DIM definierten Variablen werden automatisch auf 0 gesetzt.
DIM.. (.....)	30 DIM A(6,5)	Reserviert Speicherplätze für ein „zweidimensionales" Feld von Variablen, die die Namen A(0,0), A(0,1), ... A(0,5) A(1,0), A(1,1), ... A(6,0), A(6,1), ... A(6,5) erhalten. (Auch Matrix genannt)
DIM.. (......)	40 DIM A(8,8,8,6)	Reserviert Speicherplätze für ein mehrfach indiziertes Variablenfeld. Bei PC1500 sind höchstens zweidimensionale Felder möglich.
DIM..$ (.....)	50 DIM A$(3,4,5)	Reserviert Speicherplätze für ein mehrfach indiziertes Feld von Stringvariablen und weist allen den Leerstring " " zu. Bei einigen Computern kann ein String bis zu 255 Zeichen enthalten; bei anderen z. B. ZX 81 definiert die letzte Ziffer die gemeinsame maximale Länge aller Strings des Feldes.
	Auch möglich: 5 CLEAR 10 INPUT I 20 DIM A(I) ... 80 GOTO 5	Im DIM-Befehl dürfen auch Variable stehen. Ein Feld darf in der Regel während eines Programmlaufs jedoch nur einmal dimensioniert werden. Anderenfalls erscheint eine Fehlermeldung.
		Abhilfe: Der Befehl CLEAR löscht alle Reservierungen und ermöglicht so die nebenstehende Programmschleife.
	10 DIM A(4), B%(5), A$(7), B$(8,7)	Mehrere DIM-Befehle lassen sich zusammenfassen.

Scan: S. Höltgen

	Nützlich: 10 INPUT N 20 DIM A(N) 30 FOR I=0 TO N 40 INPUT A(I) 50 NEXT I	Dieses Programm erwartet N+1 Daten und speichert sie auf den Plätzen A(0), ... A(N).

15. Steuerbefehle

BASIC-Zeichen	Benutzungsweise	Bedeutung
RUN	RUN RUN 100	Der Befehl RUN veranlaßt den Computer, das im Arbeitsspeicher befindliche Programm, beginnend mit der niedrigsten Zeilennummer abzuarbeiten. Zuvor werden alle Speicher gelöscht. Soll ein Programm erst ab einer bestimmten Zeile laufen, so wird deren Zeilennummer dem RUN-Befehl nachgestellt.
GOTO...	GOTO 100	Wirkt wie RUN 100; jedoch werden durch diesen Befehl keine Speicher gelöscht.
LIST	LIST LIST 50 LIST – 50 LIST 50–80 LIST 20–	Das Programm im Arbeitsspeicher wird angezeigt. Es können auch einzelne Zeilen bzw. Programmteile aufgerufen werden.
NEW	NEW 5000 NEW	Löscht das Programm im Arbeitsspeicher und alle Variablen. Programm löscht sich selbst.
BREAK		Tastenbefehl, der den Programmlauf anhält. Meist wird die Stelle der Unterbrechung angegeben.
STOP	300 STOP 400 IF Z < 0 THEN STOP	BASIC-Befehl, der zum Anhalten eines laufenden Programms führt. Wird ein Programm von außen durch BREAK oder den Programmbefehl STOP unterbrochen, so können die Werte aller Variablen kontrolliert werden. (Lesen: z. B. PRINT A, Abändern: z. B. A=7).
CONT	CONT	Mit dem Befehl CONT kann das Programm von der Stelle der Unterbrechung an wiederaufgenommen werden. Nach Änderungen von Programmzeilen ist die Wiederaufnahme durch CONT unmöglich, ebenso nach eventuell aufgetretenen Fehlermeldungen.
END	1000 END	Abschließender Befehl eines Programms. Wird das Programm vor Ablauf rechnerintern aufbereitet („compiliert"), so ist dieser Befehl unerläßlich.
CLEAR (CLR)	CLEAR 100 CLEAR	Löscht die Inhalte aller Speicherplätze. Das Programm bleibt erhalten. (Kann Redimensionierungsfehler vermeiden helfen.)
HOME (CLS)	HOME 5 HOME	Für Bildschirm-Computer: Löscht den Bildschirm und setzt den Cursor in die Ausgangsstellung. Oft empfehlenswert als erster Programmschritt.
TRACE (TRACE ON)	TRACE 50 TRACE ...	Hilfsbefehle zur Fehlersuche. Bewirkt das Ausdrucken jeder durchlaufenden Zeilennummer.
NOTRACE (TRACE OFF)	300 NOTRACE	Hebt den TRACE-Befehl auf.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (10)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

16. Direkter Speicherzugriff

BASIC-Zeichen	Benutzungsweise	Bedeutung
PEEK	100 A = PEEK (36879) 120 C = PEEK (X) 130 PRINT CHR$ (PEEK (1527))	Die Speicherplätze im Computer sind numeriert. Die Zählung beginnt mit 0. Die Funktion PEEK (X) gibt den aktuellen Zahlenwert (als Dezimalzahl) des Speicherplatzes Nr. X an. Einige der Speicher erfüllen festgelegte Aufgaben, z. B. steuern sie den Grafikmodus, die Größe des Bildschirmfensters etc. Die zugehörigen Adressen sind vom Computertyp abhängig. Beispiele: Beim Gerät VC 20 bestimmt der Wert des Speicherplatzes 36879 die Farbe des Bildschirms. Beim Gerät Apple II geben Speicherplätze Nr. 36 und 37 die momentane horizontale und vertikale Position des Cursors an.
	Nützlich (FÜR APPLE II): 40 J = 0 50 IF 39-PEEK(36) < LEN (T$(J)) THEN PRINT 60 PRINT T$(J);" "; 70 J = J+1 80 GOTO 50	Ein im Feld T$ ( ) Wort für Wort gespeicherter Text wird ohne Trennung am Zeilenende ausgeschrieben. Dazu wird in Zeile 50 jeweils gefragt, ob das folgende Wort noch in die Zeile paßt. Gegebenenfalls wird in die nächste Zeile gesprungen.
POKE....	200 POKE 33,30 300 POKE 10000, PEEK (10000)+1	Der Befehl POKE X,A weist dem Speicherplatz Nr. X die Zahl A zu. A kann zwischen 0 und 255 liegen. Der Bereich von X richtet sich insbesondere nach der Größe des verfügbaren Arbeitsspeichers und hängt außerdem vom Computertyp ab.

17. Häufige Fehler

(Der Wortlaut der Fehlermeldung kann bei verschiedenen Computern variieren.)
A. Fehler bei Rechenschritten und Wertzuweisungen

BASIC Fehlermeldung	Beispiel	Erklärung
SYNTAX ERROR	10 A = 1,02 A$(7), B$(8,7)	Dezimalkomma statt Dezimalpunkt

Scan: S. Holtgen

	15 U = V : W 2ØA = SIN(B*3)+4) 25 M = 4,06	Doppelpunkt statt Schrägstrich als Divisionszeichen Eine Klammer ohne Gegenklammer 0 statt Ø
DIVISION BY ZERO ERROR	3Ø C = LOG (1) 4Ø D = 1/C	Es wurde versucht, durch Null zu dividieren.
ILLEGAL QUANTITY ERROR	5Ø A = SQR(-4)	Es wurde ein nicht zugelassenes Argument in einer Funktion verwendet.
OVERFLOW ERROR	1Ø A=2 2Ø FOR I = 1 TO 3 3Ø A=A ↑ A 4Ø NEXT I	Es wurde eine zu große oder zu kleine Zahl errechnet.
TYPE MISMATCH ERROR	1Ø LET A= "BLAU" 2Ø LET A$= BLAU	Es wurde versucht, in einer Zahlvariable einen String zu speichern. Der Computer interpretiert BLAU als Namen einer Zahlvariablen. Sie kann nicht in einer Stringvariablen gespeichert werden.
STRING TOO LONG ERROR	3Ø FOR I = 1 TO 3ØØØ 4Ø A$ = A$ + C$(I) 5Ø NEXT I	Es wurde versucht, zuviele Zeichen in einer Variablen zu speichern. Die Höchstzahl hängt vom Computertyp ab.
B. Fehler bei Verwendung von Feldern		
BAD SUBSCRIPT ERROR	1Ø DIM A(2Ø) 2Ø B=3; C=7 3Ø A(B*C) = 1	In Zeile 3Ø wird versucht, die Variable A(21) zu belegen. Das Feld A war aber nur bis zum Index (subscript) 2Ø reserviert worden.
REDIM'D ARRAY ERROR	1Ø DIM A(3Ø) 3Ø DIM A(75) 6Ø GOTO 1Ø	Aus rechnerinternen Gründen ist es meist nicht ohne weiteres möglich, ein schon dimensioniertes Feld während des Programmablaufs neu zu dimensionieren (Ausnahme: ZX 81).
		Insbesondere darf ein DIM-Befehl nicht ohne weiteres in einer Schleife stehen. So bewirkt z. B. der Sprung von Zeile 6Ø eine Fehlermeldung, sofern nicht inzwischen durch den Befehl CLEAR alle Dimensionierungen aufgehoben wurden.
C. Fehler bei Sprüngen und Schleifen		
UNDEF'D STATEMENT ERROR	1Ø ... 2Ø ... 3Ø GOTO 11 4Ø GOSUB 12 5Ø ON A GOTO 13, 14	Sprünge zu nicht existierenden Zeilennummern. Jede einzelne der drei Zeilen 3Ø, 4Ø, 5Ø ist fehlerhaft.
NEXT WITHOUT FOR ERROR	2Ø GOTO 5Ø 3Ø FOR I = 1 TO 1ØØ 4Ø A = I* SIN(I*K) 5Ø... 6Ø NEXT I	Sprung in eine FOR-NEXT-Schleife. Die Rücksprungadresse zum Schleifenbeginn konnte nicht registriert werden.
RETURN WITHOUT GOSUB ERROR	1ØØ GOTO 1ØØØ 1ØØØ ... 1Ø1Ø RETURN	Sprung in ein Unterprogramm. Es konnte keine Rücksprungadresse registriert werden. Dieser Fehler tritt insbesondere dann auf, wenn das Unterprogramm am Programmende steht und der Rechner nach Abarbeiten des Hauptprogramms nicht, z. B. durch STOP, daran gehindert wird, schließlich in das Unterprogramm zu laufen.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (11)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

17. Häufige Fehler

BASIC Fehlermeldung	Beispiel	Erklärung
	D. Weitere Syntaxfehler	
SYNTAX ERROR	1Ø LET A> = 1Ø	Nicht existierender Befehl. (A> = 1Ø ist eine mathematische Aussage, kein Befehl.)
BAD SUB-SCRIPT ERROR	2Ø X = ARC TG(15)	Der Computer interpretiert ARCTG als Namen eines Feldes, das jedoch nicht dimensioniert wurde. Gemeint war der Wert der Funktion ATN(15).
UNDEF'D FUNC-TION ERROR	3Ø C = FN A(4) 4Ø DEF FN A(X) = X↑3-X	Falsche Reihenfolge: Die Definitionszeile muß vor der ersten Benutzung einer selbstdefinierten Funktion auftreten.
SYNTAX ERROR	5Ø LET MON = 1	Es wurde in einem Variablennamen ein reserviertes Wort (ON) benutzt.
	E. Eingabefehler	
REDO FROM START BZW. REENTER	1Ø INPUT A EINGABE: B	Eine Zahl wurde angefordert, ein Textzeichen eingegeben. Der Computer fordert erneut eine Zahl an.
EXTRA IGNORED	1Ø INPUT A, B, C EINGABE: 1Ø, 2Ø, 3Ø, 4Ø	Es wurden mehr Werte eingegeben als angefordert. Der Computer ignoriert überzählige Daten.
??	1Ø INPUT A, B, C EINGABE: 1Ø, 11	Es wurden weniger Daten als erwartet eingegeben. Der Computer fordert die restlichen an.
	F. Einige typische Programmierfehler	
	1ØØ DEF FN F(X)=X↑.2Ø5 2ØØ Y = F(3)	Nicht die Funktion FN F wird aufgerufen, sondern das Feld F. Da manche Computer es zulassen, auch mit nicht vorher dimensionierten Feldern zu arbeiten, wird in Zeile 2ØØ der Wert des Feldes auf den Speicherplatz Y geschrieben. (Z. B. APPLE II: Felder mit maximalem Index 1Ø benötigen keine vorhergehende DIM-Anweisung.) Eine Fehlermeldung kommt durch Zeilen 1ØØ und 2ØØ nicht zustande.
	2O A = 4∗B	Anstelle der Ziffer Ø wurde bei der Zeilennummer der Buchstabe O benutzt. Es entsteht die Programmzeile 2, in der der Variablen OA der Wert 4∗B zugewiesen wird. Die eventuell

Scan: S. Höltgen

		vorher vorhandene Programmzeile 2 wird gelöscht, es tritt keine Fehlermeldung auf.
	5Ø VERLUST =PREIS/1.13 6Ø VC=2∗Z : VD=L∗L 7Ø VE= VD-VC∗VD	Bei vielen Computern werden bei Variablennamen nur die ersten zwei Zeichen beachtet, so daß VERLUST und VE denselben Speicherplatz bezeichnen. Das Identifizieren als verschieden gemeinter Speicherplätze durch den Computer kann zu unsinnigen Ergebnissen führen.

18. Speichern von Programmen auf Cassette oder Diskette

Befehl	Benutzungs-hinweise	Bedeutung
SAVE (CSAVE)	SAVE "ZINSEN" SAVE ZINSEN SAVE	Das Programm im Arbeitsspeicher wird auf Kassette oder Diskette zu späterem Gebrauch gespeichert. Es besteht i. a. die Möglichkeit, ihm einen Namen zu geben, unter dem es wieder aufgerufen werden kann. Das ist insbesondere dann wünschenswert, wenn verschiedene Programme auf einem Band gespeichert werden sollen: bei Disketten ist die Benennung des Programms unerläßlich.
		Für die Namen von Programmen gelten folgende Richtlinien, die allerdings vom Computertyp abhängen können: 1. Ein Name darf nicht mit einer Ziffer beginnen. 2. Die Länge eines Namens ist begrenzt. (Z. B. 3Ø Zeichen bei APPLE II) 3. Das Sonderzeichen , darf nicht Bestandteil eines Namens sein. 4. Eventuell muß der Name in Anführungszeichen gesetzt werden. 5. In manchen Fällen können mit Namen versehene Programme nur auf Diskette gespeichert werden.
		Bei Verwendung eines Kassettengeräts muß der Benutzer darauf achten, daß – das Gerät während der Aufnahme läuft, – mit der richtigen „Lautstärke" aufgenommen bzw. wiedergegeben wird, – das Programm an einer freien Stelle des Bandes gespeichert wird; andernfalls werden vorhandene Programme überschrieben. Diese Maßnahmen werden bei Benutzung einer Diskettenstation vom Computer selbst vorgenommen. (**D**isk **O**perating **S**ystem = DOS)
VERIFY (CLO-AD?)	VERIFY "ZINSEN" VERIFY	Bei Kassetten: Der Computer sucht nach einem Programm mit dem angegebenen Namen und vergleicht es mit dem Programm im Arbeitsspeicher. Das Resultat (Übereinstimmung oder nicht) wird angezeigt. VERIFY ohne Namen veranlaßt den Vergleich des Arbeitsspeichers mit dem nächsten Programm auf der Kassette. Bei Disketten: Es wird oft nur geprüft, ob ein Programm oder eine Datei mit dem angegebenen Namen vorhanden ist.

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Scan: S. Höltgen

Stichwort: BASIC (12)

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

18. Speichern von Programmen auf Cassette oder Diskette

Befehl	Benutzungs-hinweise	Bedeutung
SAVE (CSAVE)	SAVE "ZINSEN" SAVE ZINSEN SAVE	Das Programm im Arbeitsspeicher wird auf Kassette oder Diskette zu späterem Gebrauch gespeichert. Es besteht i. a. die Möglichkeit, ihm einen Namen zu geben, unter dem es wieder aufgerufen werden kann. Das ist insbesondere dann wünschenswert, wenn verschiedene Programme auf einem Band gespeichert werden sollen: bei Disketten ist die Benennung des Programms unerläßlich.
VERIFY (CLOAD?)	VERIFY "ZINSEN" VERIFY	Bei Kassetten: Der Computer sucht nach einem Programm mit dem angegebenen Namen und vergleicht es mit dem Programm im Arbeitsspeicher. Das Resultat (Übereinstimmung oder nicht) wird angezeigt. VERIFY ohne Namen veranlaßt den Vergleich des Arbeitsspeichers mit dem nächsten Programm auf der Kassette. Bei Disketten: Es wird oft nur geprüft, ob ein Programm oder eine Datei mit dem angegebenen Namen vorhanden ist.
LOAD (CLOAD)	LOAD "ZINSEN" LOAD	Der Computer sucht auf der Kassette bzw. der Diskette nach einem Programm mit dem betreffenden Namen und lädt es in den Arbeitsspeicher. LOAD ohne Namen bewirkt das Laden des ersten auf der Kassette gefundenen Programms. Ein vorher im Arbeitsspeicher befindliches Programm geht verloren.
MERGE		Nur bei einigen Computern vorhanden: Es ist u. U. möglich, einem bereits im Arbeitsspeicher vorhandenen Programm ein anderes auf Kassette oder Diskette gespeichertes Programm anzufügen. (Beispiel: Hilfsroutinen und Unterprogramme, die häufig in verschiedenem Zusammenhang benötigt werden.)

Scan: S. Höltgen

Befehl	Benutzungs-hinweise	Bedeutung
RUN	RUN "ZINSEN"	Das genannte Programm wird gesucht, geladen und sofort ausgeführt.
DELETE (PURGE)	DELETE "ZINSEN"	Nur bei Disketten: Das betreffende Programm wird auf der Diskette gelöscht, sofern es nicht geschützt ist.
LOCK UNLOCK	LOCK "ZINSEN" UNLOCK "ZINSEN"	Schützen eines Programms gegen Löschung durch DELETE; Aufheben des Schutzes.
OPEN WRITE CLOSE READ		Dateien: Es ist möglich, beliebige Daten wie etwa Meßwerte, Anschriften, Standardbrieftexte etc. auf Kassette oder Diskette zu speichern. Dazu wird eine Datei eröffnet (OPEN), die Daten werden eingeschrieben (WRITE) und die Eingabe wird abgeschlossen (CLOSE). Später wird die Information nach Bedarf ausgelesen (READ) und im Computer weiterverarbeitet. Die Einzelheiten der Datei-Befehle Ihres Computers entnehmen Sie dem Benutzerhandbuch.

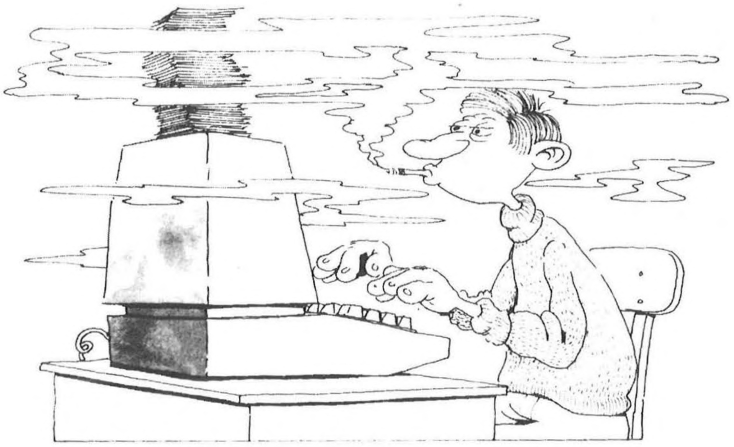
19. Spezielle Befehle Ihres Computers

Darüber hinaus gibt es Befehle, die sich auf Ihren speziellen Computer beziehen. Vor allem die folgenden Themen sind häufig von Wichtigkeit.

1. Grafik-Befehle, Bildschirm-Koordinaten, Farbwahl
2. Erzeugung von Tönen
3. Einsatz von Handsteuergeräten (Paddles, Joystick etc.)
4. Eröffnen, Redigieren und Lesen von Dateien
5. Reservierte Wörter
6. Bedeutung und Plazierung von Systemvariablen
7. Aufruf der Maschinensprache

Aus Weber-Mrowka, Grundkenntnisse BASIC, Giradet-Verlag, Essen, 5,80 Mark

Die BASIC-Stichwörter sind damit beendet. Ab der nächsten Ausgabe erhalten Sie an dieser Stelle eine Programmbibliothek zum Sammeln.



Scan: S. Höltgen

Stichwort: MSX

Die wichtigsten Begriffe aus der Computer-Technik – in Stichworten zusammengefaßt

Ein typischer Vertreter der MSX-Rechner

MSX ist die Abkürzung für ,,Microsoft Super Extended". Microsoft ist einer der weltweit führenden Software-Hersteller, dessen Betriebssystem MS-DOS (Microsoft Disk Operating System) auch im Bereich der Personal-Computer mittlerweile zu einem Standard geworden ist.

Der hauptsächliche Anlaß für die Entwicklung des MSX-Standards war der, dem Home-Computer-Anwender das Schicksal zu ersparen, mit dem Kauf eines Geräts auf einen ganz bestimmten Hersteller festgelegt zu sein. Tatsächlich ist es bisher so, daß man auf seinem Gerät nur die speziell dafür entwickelte Software laufen lassen kann. Aber auch der Anschluß von Peripheriegeräten, etwa Drucker und externe Massenspeicher, ist problematisch: Entweder es geht überhaupt nicht, oder entsprechende zusätzliche Schnittstellen müssen für meist teures Geld dazugekauft werden.

Zwei Fliegen mit einer Klappe

Der in Japan entwickelte und dort bereits weit verbreitete MSX-Standard schlägt beide Fliegen mit einer Klappe. Einmal sind das MSX-Betriebssystem und die Programmiersprache MSX-BASIC so vereinheitlicht, daß alle für MSX geschriebenen Programme auf allen Home-Computern laufen, die sich dem MSX-Standard unterworfen haben. Zum zweiten wurden

Scan: S. Höltgen

auch die Peripherie-Schnittstellen so normiert, daß zum Beispiel Drucker und Diskettenlaufwerke untereinander austauschbar sind. Bei der Druckerschnittstelle hat man sich bei MSX zudem auf einen weit verbreiteten Industriestandard eingestellt: den parallelen Centronics-Standard, mit dem heute fast alle gängigen Druckertypen ausgerüstet sind.

Enorme Menge an Programmen

Aber die Standardisierung geht noch weiter. Zum Beispiel durch das einheitliche Bildschirmformat und die dazugehörige, sehr wichtige Bildschirmsteuerung. Zu nennen ist der einheitliche Steckplatz für fertige Software-Module, so daß bei Verwendung dieser Art von Programmen das zeitraubende Laden über ein Kassetten- oder Diskettenlaufwerk entfällt. Und es existieren auch einheitliche Anschlußstellen für die bekannten ,,Freuden-Knüppel", die Joysticks, die eine äußerst einfache Steuerung besonders von Computerspielen erlauben.

Wohl die wichtigste gemeinsame Eigenschaft von MSX-Computern betrifft schließlich eine bedeutende Erweiterungsmöglichkeit. Während man bei den meisten bisherigen Home-Computern nämlich tatsächlich auf die eher spielerische Anwendung im privaten Heimbereich festgelegt war, bringt der MSX-Standard von Haus aus viele wichtige Voraussetzungen auch für eine spätere professionelle Nutzung mit. Die wichtigste dieser Voraussetzungen ist die CP/M-Fähigkeit von MSX-Computern. CP/M ist die Abkürzung von ,,Control Program for Microcomputers". Dies ist ein weit verbreitetes Betriebssystem, für das eine große Zahl von äußerst nützlichen Software-Produkten angeboten wird. Als wichtigste Typen von derartigen Programmen sind vor allem die Textverarbeitung und Kalkulation zu nennen. Darüber hinaus gibt es aber auch viele Spezialprogramme unter CP/M, die im allgemeinen unter der Bezeichnung Branchen-Software gehandelt wird. Damit werden MSX-Computer auch für kommerzielle Anwendungen nutzbar.

Vom Einsteiger zum Profi

Diese Möglichkeiten dürften für viele Anwender von großem Interesse sein, die in die elektronische Datenverarbeitung zunächst einmal ,,hineinriechen" möchten, ohne sich gleich auch in größere finanzielle Unkosten stürzen zu müssen. Später dann, wenn sie mit dem neuen Medium Computer umgehen gelernt und seine Möglichkeiten erkannt haben, können sie dann den Schritt in die professionelle Nutzung des Computers tun, ohne daß ein kompletter Neuanfang mit einem dafür geeigneten Gerät nötig ist.

Scan: S. Höltgen

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann. Teil 1

Sortieren von Daten (A)

```
 100 REM INPUT
 120 Z=12
 140 DIM S$(Z)
 160 DATA MOELLER KLAUS,SCHULZ ARNDT
 180 DATA ZERBLER KLAUS,ALNO TOM
 200 DATA ZERBST KLAUS, AALBOHM KURT
 220 DATA SCHOLZ OTTO,ALNO UWE
 240 DATA SCHULZE ARNO,SCHOLZ KURT
 260 DATA SCHULTZ FRED,SCHULZE BERT
 500 GOSUB 1000
 600 REM OUTPUT
 620 PRINTCHR$(13)
 640 FOR I=1 TO Z
 660 PRINT S$(I):NEXT
 680 END
 980 REM
 990 REM ROUTINE: SORTIEREN (A)
1000 FOR I=1 TO Z
1005 READ S$(I):NEXT I
1010 FOR I=1 TO Z-1
1015 FOR J=I+1 TO Z
1020 IF S$(I)<=S$(J) THEN 1035
1025 S$=S$(I):S$(I)=S$(J)
1030 S$(J)=S$
1035 NEXTJ
1040 NEXTI
1045 RETURN
```

Der Aufruf

Das Unterprogramm von Zeile 1000 bis Zeile 1045 erzeugt eine sortierte Liste S$(I). Die Zeilen 100 bis 260 enthalten die Anzahl der Daten Z (Zeile 120), die DIM-Anweisung (Zeile 140) und die unsortierten Daten (Zeilen 160 bis 260). Diese müssen dem Unterprogramm übergeben werden. Die vom Unterprogramm erzeugte sortierte Liste S$(I) wird in den Zeilen 620 bis 680 auf den Bildschirm ausgegeben. Ist eine nichtsortierte Liste S$(I) bereits vorhanden, so erzeugt das gleiche Unterprogramm ohne die Zeilen 1000 und 1005 eine sortierte Liste. Die ursprüngliche Liste geht dabei „verloren".

Die Routine

In den Zeilen 1000 und 1005 werden die DATA-Zeilen in die nichtsortierte Liste S$(I) eingelesen. In zwei Schleifen (Zeilen 1010 bis 1040) erfolgt das Sortieren nach folgendem Schema: Mit I = 1 wird in der äußeren Schleife der erste Name S$(1) gewählt und mit allen restlichen – S$(2) bis S$(Z) – in der inneren Schleife verglichen. Ist S$(1) kleiner (oder gleich) als

Scan: S. Höltgen

alle restlichen (Zeile 1020), dann bleibt S$(1) an erster Position. Ist S$(1) größer, dann erfolgt in den Zeilen 1025 und 1030 ein Ringtausch über die Variable S$. Auf diese Weise enthält S$(1) auf jeden Fall den kleinsten Wert des Feldes. Mit I = 2 wird danach der Vergleich in analoger Weise fortgesetzt.

Sortieren von Daten (B)

```
 100 REM INPUT
 120 Z=12
 140 DIM N$(Z),S$(Z)
 160 DATA MOELLER KLAUS,SCHULZ ARNDT
 180 DATA ZERBLER KLAUS,ALNO TOM
 200 DATA ZERBST KLAUS, AALBOHM KURT
 220 DATA SCHOLZ OTTO,ALNO UWE
 240 DATA SCHULZE ARNO,SCHOLZ KURT
 260 DATA SCHULTZ FRED,SCHULZE BERT
 500 GOSUB 1000
 600 REM OUTPUT
 620 PRINTCHR$(13)
 640 FOR I=1 TO Z
 660 PRINT S$(I):NEXT
 680 END
 980 REM
 990 REM ROUTINE: SORTIEREN (B)
1000 FOR I=1 TO Z:READ N$(I)
1005 S$(I)="":NEXT I:J=0
1010 FOR I=1 TO Z
1015 IF S$(I)="" THEN 1035
1020 IF S$(I)<=N$(J) THEN 1045
1025 FOR T=Z TO I+1 STEP -1
1030 S$(T)=S$(T-1):NEXT T
1035 S$(I)=N$(J):J=J+1:I=0
1040 IF J=Z+1 THEN RETURN
1045 NEXT I
```

Der Aufruf

Das Unterprogramm von Zeile 1000 bis Zeile 1045 erzeugt eine unsortierte Liste N$(I) und eine sortierte Liste S$(I). Beide Listen stehen zur Verfügung. Dem Unterprogramm werden die unsortierten Daten (Zeilen 160 bis 260), deren Anzahl Z (Zeile 120) und die DIM-Anweisung für die Listen (Zeile 140) übergeben. Die vom Unterprogramm erzeugte Liste S$(I) wird in den Zeilen 620 bis 680 ausgegeben.

Die Routine

In Zeile 1000 werden die zu sortierenden Daten in das Feld N$(I) eingelesen und die Namen der Liste S$(I) auf Leerstring (Zeile 1005) gesetzt. Die Variable J zählt die unsortierten Daten. Sind alle eingegebenen Daten sortiert, so erfolgt aus Zeile 1040 der Rücksprung. Der Sortiervorgang erfolgt in der Schleife (Zeilen 1010 bis 1045). Ein Name N$(J) wird mit der sortierten Tabelle S$(I) verglichen. Ist die Stelle S$(I) leer (Zeile 1015), so wird einsortiert (Zeile 1035) und der nächste Begriff (J = J+1) geholt. Ist S$(I) nicht leer und ist N$(J) kleiner als S$(I), dann werden alle nachfolgenden Namen in der sortierten Liste um einen Platz höher gesetzt (Schleife in den Zeilen 1025 bis 1030) und an dieser Stelle der neue Begriff eingefügt.

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

Scan: S. Höltgen

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 2)

3. Umwandlung von Dezimalzahlen

Der Aufruf
Das Unterprogramm ab Zeile 1000 erzeugt eine codierte Zahl für ein Zahlensystem, dessen Basis zwischen 2 und 10 einschließlich liegt. Dem Unterprogramm müssen die Dezimalzahlen N% (Zeile 140) und die Basis S des Zahlensystems (Zeile 160) übergeben werden. Die vom Unterprogramm erzeugte codierte Zahl wird in Zeile 720 als Stringvariable ausgegeben.
Die Routine
In Zeile 1000 werden die Anfangswerte gesetzt. In den Zeilen 1010 und 1015 wird die höchste Potenz X ermittelt, die als Schleifenvariable in Zeile 1025 benutzt wird. In Zeilen 1030 und 1040 erfolgt die Umrechnung. Die Darstellung der codierten Zahl geschieht in den Zeilen 1035 und 1050.

4. Umwandlung von codierten Zahlen

Der Aufruf
Das Unterprogramm ab Zeile 1000 wandelt eine codierte Zahl des Zahlensystems mit der Basis S in eine Dezimalzahl um. Die codierte Zahl darf nur Ziffern enthalten, die kleiner als die Basiszahl S sind. Dem Unterprogramm werden die codierte Zahl als Stringvariable in Zeile 180 und das Zahlensystem S in Zeile 140 übergeben. Die Ausgabe der Dezimalzahl erfolgt in Zeile 740. Die Zeilen 720 und 780 geben eine Fehlermeldung aus, wenn die codierte Zahl nicht zum Zahlensystem der Basis S gehört.
Die Routine
In den Zeilen 1005 und 1010 wird die Länge der codierten Zahl — nicht größer als 80 Zeichen — überprüft. In den Zeilen 1020, 1025 und 1035 wird auf korrekte Codeeingabe kontrolliert. Ist der eingegebene Code falsch, wird das Flag F gesetzt und aus dem Unterprogramm zurückgesprungen (Zeile 1040). In den Zeilen 1050 und 1055 erfolgt die Umrechnung in die Dezimalzahl innerhalb der Schleife bei korrekter Eingabe.

Scan: S. Höltgen

```
100   REM INPUT
120   PRINT CHR$(147);
140   INPUT "ZAHL";N%
160   INPUT "ZAHLENSYSTEM";S
180   IF S<2 OR S>10 THEN 160
500   GOSUB 1000
700   REM OUTPUT
720   PRINT SPC(2);A$
760   GOTO 140 : REM END
980   REM
990   ROUTINE: UMWANDLUNG VON
      DEZIMALZAHLEN
1000  X=-1 : A$="" : B$=""
1005  REM ANZAHL DER STELLEN
1010  X=X+1
1015  IF S↑X<N% THEN1010
1020  REM UMRECHNUNG
1025  FOR I=X TO 0 STEP -1
1030  K%=S↑I : B=INT(N%/K%)
1035  B$=MID$(STR$(B),2,1)
1040  IF B< >0 THEN N%=N%-B * K%
1045  REM ZAHL DARSTELLEN
1050  A$=A$+B$
1055  NEXT I
1060  RETURN
```

```
100   REM INPUT
120   PRINT CHR$(147);
140   INPUT" ZAHLENSYSTEM";S
160   IF S<2 OR S>10 THEN 140
180   A$="" : INPUT" ZAHL";A$
500   GOSUB 1000
700   REM OUTPUT
720   IF F=1 THEN 780
740   PRINT" DEZIMALZAHL";Z
760   GOTO 140 : REM END
780   PRINT " FALSCHE ZAHL !" : END
980   REM
990   REM ROUTINE: UMWANDLUNG VON
      CODIERTEN ZAHLEN
1000  REM STRINGLAENGE PRUEFEN
1005  Z=0 : L=LEN(A$)
1010  IF L>80 THEN GOTO 1060
1015  FOR I=L TO 1 STEP-1
1020  B$=MID$(A$,I,1)
1025  X%=VAL(B$) : F=0
1030  REM CODE PRUEFEN
1035  IF X%>S OR X%<0 THEN F=1
1040  IF F=1 THEN RETURN
1045  REM UMRECHNEN
1050  Y%=X% * S↑(L-I)
1055  Z=Y%+Z : NEXT I
1060  RETURN
```

Die Programmbeispiele laufen ohne Änderungen auf den Commodore-Home-Computern, für andere können Anpassungen nötig sein.

Scan: S. Höltgen

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 3)

5. Teiler einer natürlichen Zahl

```
 100 REM INPUT
 120 PRINT CHR$(147)
 140 DIM T(100)
 160 PRINT" NATUERLICHE ZAHL?"
 180 OPEN1,0
 200 PRINT SPC(2);
 220 INPUT#1,A$
 240 N=VAL(A$)
 260 CLOSE1
 500 GOSUB 1000
 700 REM OUTPUT
 720 PRINT
 740 FOR J=1 TO I STEP 2
 760 PRINT T(J); T(J+1)
 780 NEXTJ: GOTO160
 980 REM
 990 REM ROUTINE: TEILER
1000 REM EINGABE PRUEFEN
1005 T(1)=1: T(2)=N: I=2
1010 IF N<>INT(N) THEN1060
1015 IF N<1 THEN 1060
1020 IF N<4 THEN RETURN
1025 REM TEILER ERMITTELN
1030 X=1: Y=SQR(N): I=0
1035 X=X+1: IF X>Y THEN RETURN
1040 Z=N/X
1045 IF Z<>INT(Z) THEN 1035
1050 I=I+1: T(I)=X: I=I+1
1055 T(I)=Z: GOTO 1035
1060 PRINT
1065 PRINT" KEINE NATUERLICHE"
1070 PRINT" ZAHL!!!"
1075 I=0:RETURN
```

Der Aufruf

Das Unterprogramm ab Zeile 1000 ermittelt die Teiler einer natürlichen Zahl. Die Teiler werden in einem eindimensionalen Feld T(I) gespeichert. Dem Unterprogramm müssen die DIM-Anweisung (Zeile 140) und die natürliche Zahl N (Zeile 240) zur Verfügung gestellt werden. Das Unterprogramm erzeugt die unsortierte Liste T(I) der Teiler. Die Ausgabe des eindimensionalen Feldes T(I) der Teiler erfolgt in den Zeilen 700 bis 780, wobei zwei nebeneinanderstehende Zahlen als Produkt die eingegebene Zahl N ergeben.

Scan: S. Höltgen

Die Routine

In Zeile 1005 werden die Primzahlbedingungen gesetzt. Zeilen 1010 und 1015 überprüfen die Eingabe. Bei fehlerhafter Eingabe erfolgt in den Zeilen 1060–1075 Anzeige und Rücksprung. Für die Berechnung der Teiler werden in Zeile 1030 die Anfangs- (X = 1; I = 0) und Endbedingungen (Y = SQR(N)) gesetzt. In den Zeilen 1035 bis 1045 erfolgt eine fortlaufende Division und die Abfrage, ob das Ergebnis der Division ganzzahlig ist. Ist dies der Fall, dann wird in den Zeilen 1050 bis 1055 das Ergebnis der Division im Feld T(I) gespeichert.

6. Kleinstes gemeinsames Vielfaches

```
 100 REM INPUT
 120 PRINT CHR$(147)
 160 PRINT" ZWEI NATUERLICHE"
 180 PRINT" ZAHLEN EINGEBEN!"
 200 OPEN1,0
 220 PRINT SPC(2);
 240 INPUT#1,N1
 260 PRINT:PRINTSPC(2)
 280 INPUT#1,N2
 300 CLOSE1
 500 GOSUB 1000
 700 REM OUTPUT
 720 PRINT
 740 PRINT" DAS KLEINSTE GEMEIN-
 760 PRINT" SAME VIELFACHE
 780 PRINT" IST:":PRINTSPC(2)KG
 800 GOTO 160
 980 REM
 990 REM ROUTINE: KGV
1000 REM KLEINSTE GEMEINSAME
1005 REM VIELFACHE ERMITTELN
1010 N=1: M=1
1015 S=N1*M: T=N2*N
1020 IF S=T THEN KG=S: RETURN
1025 IF S>T THEN N=N+1: GOTO 1015
1030 IF T>S THEN M=M+1: GOTO 1015
```

Der Aufruf

Das Unterprogramm ab Zeile 1000 ermittelt das kleinste gemeinsame Vielfache (kgV; Hauptnenner) zweier natürlicher Zahlen. Dem Unterprogramm werden die Zahlen N1 und N2 übergeben (Zeilen 240 bzw. 280). Das Unterprogramm erzeugt das kgV in der Variablen KG und die Vielfachen der Zahlen N1, N2 in den Variablen N, M. In Zeile 780 wird das kleinste gemeinsame Vielfache KG ausgegeben.

Die Routine

In Zeile 1010 werden N = 1, M = 1 (die Vielfachmultiplikation) auf die Anfangswerte gesetzt. In den restlichen Zeilen erfolgen die Bildung der Vielfachen (Zeile 1015) und ein Vergleich. Bei Übereinstimmung der Vielfachenprodukte wird das Unterprogramm in Zeile 1020 verlassen.

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

Scan: S. Höltgen

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 4)

7. Berechnung von Speicherinhalten

Der Aufruf

Das Unterprogramm ab Zeile 1000 berechnet den Inhalt von Speicherstellen im Binärcode und außerdem als Sedezimalzahl (Hexcode). Dem Unterprogramm muß die Speicheradresse in der Variablen S übergeben werden.
Durch Ändern der Zeile 1010 (N = S) kann das Unterprogramm auch für beliebige Zahlenumwandlungen benutzt werden. Die erzeugte Binär- und Sedezimalzahl werden in den Variablen Q$ und P$ in Zeile 740 bzw. 760 ausgegeben.

Die Routine

Zeile 1005 setzt die Anfangsbedingungen. Zeile 1010 ermittelt den Inhalt N der Speicheradresse S. In der nachfolgenden Schleife (Zeile 1015 bis 1035) wird der Wert N durch einen logischen Vergleich in Zeile 1025 in die Binärzahl P$ (Zeile 1030) umgerechnet.
Zeile 1040 – 1055 berechnet aus dem Speicherinhalt N die Sedezimalzahl Q$.
Die restlichen Zeilen erzeugen aus dem Speicherinhalt N einen linksbündigen String N$.

8. Periode von Brüchen

Der Aufruf

Das Unterprogramm ab Zeile 1000 erzeugt für echte Brüche (Zähler kleiner als der Nenner) eine Dezimalzahl mit maximal 255 Stellen. Damit kann die Periode von Bruchzahlen bestimmt werden. Dem Unterprogramm müssen die Zahlenwerte für Zähler, Nenner und die Länge der Dezimalzahl in den Variablen Z, N und L übergeben werden. Die Dezimalzahl wird in der Stringvariablen Q$ erzeugt und danach in Zeile 740 ausgegeben.

Scan: S. Höltgen

Die Routine

Zeile 1005 und 1010 dienen der Vorbereitung für die Berechnung der Dezimalzahl. Zeile 1020 prüft, wie oft der Nenner N im zehnfachen Wert des Zählers Z enthalten ist. Diese Zahl wird in der Variablen P gespeichert. Zeile 1035 erzeugt aus der Variablen P den String P$.
Zeile 1040 erzeugt die Dezimalzahl Q$ aus der Variable P$ solange, bis die Bedingung in Zeile 1045 (maximale Stellenzahl) erfüllt ist.

```
 100 REM INPUT
 180 PRINT
 200 INPUT "ADRESSE"; S
 220 IF S<0 OR S>65535 GOTO 200
 240 PRINT
 500 GOSUB 1000
 700 REM OUTPUT
 720 PRINT SPC(1) N$;
 740 PRINT SPC(2) P$;
 760 PRINT SPC(2) Q$
 780 PRINT
 800 GOTO 200
 980 REM
 990 REM ROUTINE: SPEICHERINHALT
1000 P$=" ":M$=" ":N$=" ":Q$" "
1010 N=PEEK(S)
1015 FOR I=7 TO 0 STEP-1
1020 K=2↑I
1025 P$(I)=STR$(ABS((N AND K)=K))
1030 P$=P$+RIGHT$(P$(I),1)
1035 NEXT I
1040 A=INT(N/16): B=N-A*16
1045 F=48: IF A>9 THEN F=55
1050 G=48: IF B>9 THEN G=55
1055 Q$=CHR$(A+F)+CHR$(B+G)
1060 N$=STR$(N)
1065 L=LEN(N$)
1070 FOR I=L TO 4
1075 M$=M$+" ": NEXT I
1080 N$=M$+N$
1085 RETURN
```

```
 100 REM INPUT
 180 PRINT
 200 INPUT "ZAEHLER"; Z
 220 INPUT "NENNER"; N
 240 IF Z> =N GOTO 180
 260 INPUT "LAENGE"; L
 280 IF L<3 OR L>255 GOTO 260
 500 GOSUB 1000
 700 REM OUTPUT
 720 PRINT
 740 PRINT SPC(6)K": "N"="
 760 PRINT Q$
 780 GOTO 180
 980 REM
 990 REM ROUTINE: PERIODE
1000 P$=" ": Q$="0.": K=Z
1010 P=-1: Z=Z*10
1015 P=P+1
1020 Y=Z-(P+1)*N
1025 IF Y<0 GOTO 1035
1030 GOTO 1015
1035 P$=STR$(P)
1040 Q$=Q$+RIGHT$(P$,1)
1045 IF LEN(Q$)=L THEN RETURN
1050 Z=Y+N: GOTO 1010
```

Die Programmbeispiele laufen ohne Änderungen auf den Commodore-Home-Computern, für andere können Anpassungen nötig sein.

Scan: S. Höltgen

Spectrum: Zwischen den Zeilen

Das Programm ist in Maschinencode für den Spectrum 48K geschrieben und liegt in DATA-Anweisungen vor. Nach seiner Generierung kennt der Computer einen neuen PRINT AT-Befehl. Mit ihm kann man beliebig zwischen den Zeilen und Spalten drucken. Es ist möglich, alte Zeichen teilweise zu überschreiben. Damit sind leicht Pseudo-3D-Effekte und andere Schriftgrafiken programmierbar. Man kann auch Laufschriften erzeugen, die sich **pixelweise** bewegen. Mit einer einzigen BASIC-Zeile lassen sich 36 Zeichen pro Zeile realisieren.

Zur Inbetriebnahme

Tippen Sie das Listing ein und sichern Sie es auf Band (wichtig, falls zunächst Fehleingaben in den DATA-Zeilen gemacht wurden). Starten Sie das Programm mit RUN und speichern Sie danach den gewonnenen Objektcode mit SAVE "PRINT #7" CODE 59986,280 ab. Fortan müssen Sie nur dieses Programm laden und den Befehl initialisieren, um über ihn verfügen zu können. Vor dem ersten Benutzen (nach dem Laden oder jedem NEW) muß der neue Befehl mit RANDOMIZE USR 59993 initialisiert werden. Vergißt man dies, so gibt der Computer die Fehlermeldung "0 INVALID STREAM" aus.

Der Befehl hat das Format
PRINT #7;AT X,Y; Z$

X und Y geben die Position eines Pixelpunktes auf dem Bildschirm an. Sie entsprechen in ihrem Format den Plotkoordinaten. X kann also durch eine Variable, eine Zahl oder einen arithmetischen Ausdruck ersetzt werden. X darf aber nur die Werte 0 bis 255 einschließlich annehmen. Für Y gilt dasselbe wie für X, nur darf es nicht größer als 176 werden. Z$ steht für eine beliebige einfache Stringvariable. Ein String (in Anführungszeichen) darf nicht eingesetzt werden. X und Y geben die Position der linken unteren Ecke von Z$ an. Der Strichpunkt vor Z$ kann nicht durch ein Komma ersetzt werden. PRINT#7 funktioniert nicht mit TAB und arbeitet mit dem Drucker nur über COPY zusammen.

Tippen Sie folgendes Programm ein und starten es mit RUN:

```
10 LET A$=,,HERR"
20 FOR I=0 TO 255: PAUSE 1
30 PRINT#7; AT I,100;A$: NEXT I
```

Man kann auch Wörter in Kreisen bewegen:

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 5)

9. Fakultät

```
180   INPUT" ZAHL"; N
200   PRINT
500   GOSUB 1000
720   PRINT
740   PRINT N; CHR$(157)"!=";
760   PRINT P$: PRINT
780   GOTO 180
1000  REM ROUTINE: FAKULTAET
1005  P=1
1010  P$=STR$(P)
1015  FOR I=1 TO N
1020  L=LEN(P$): D=0
1025  FOR J=L TO 1 STEP -1
1030  X=VAL(MID$(P$,J,1))
1035  X=X*I+D
1040  D=INT(X/10)
1045  Y$=STR$(X)
1050  IF J=1 THEN 1065
1055  B$=RIGHT$(Y$,1)+B$
1060  GOTO 1070
1065  D$=Y$+B$:B$=
1070  NEXT J
1075  Q=VAL(LEFT$(D$,1)): R=LEN(D$)
1080  IF Q=0 THEN P$=RIGHT$(D$, R-1)
1085  IF Q>0 THEN P$=D$
1090  NEXT I
1095  RETURN
```

Der Aufruf

Das Unterprogramm ab Zeile 1000 ermittelt für eine natürliche Zahl n den Wert n! (gelesen: n Fakultät). Dieser Zahlenwert n! ist die Anzahl aller möglichen Anordnungen von n unterschiedlichen Objekten (Anzahl der Permutationen). n! ist eine Abkürzung für das Produkt 1 * 2 * 3 * . . . * n. Dem Unterprogramm wird die natürliche Zahl n (Zeile 180) übergeben. Die Anzahl der Permutationen wird berechnet und in der Variablen P gespeichert. Die Ausgabe dieses Wertes P erfolgt in Zeile 760.

Die Routine

Die äußere Schleife (Zeile 1015 und Zeile 1090) ergibt den jeweiligen Multiplikator I, der in jedem Durchlauf der Schleife um 1 erhöht wird. Das Produkt

Scan: S. Höltgen

selbst wird in der Stringvariablen P$ (Zeile 1065) erzeugt und gespeichert. Die Multiplikation erfolgt Stelle um Stelle in der inneren Schleife (Zeile 1025–Zeile 1070). Die Variable X (Zeile 1030) enthält den Zahlenwert der zu multiplizierenden Stelle. In Zeile 1035 erfolgt die Multiplikation, wobei ein Übertrag D aus der vorhergehenden Multiplikation dazu addiert wird. In Zeile 1040 wird der neue Übertrag für die nachfolgende Stelle berechnet. Zeile 1055 ermittelt die neue Zahl für die multiplizierte Stelle. In den Zeilen 1075 bis 1085 werden vorangestellte Nullen beseitigt.

10. Binomialkoeffizienten

```
160   PRINT CHR$(147)
180   INPUT" GESAMTZAHL";N
200   INPUT" TEILZAHL";M
500   GOSUB 1000
720   IF Q=0 THEN 760
740   PRINT:PRINT Q
760   PRINT
780   GOTO 180
1000  REM ROUTINE: BINOMIALKOEFFIZIENTEN
1005  Q=0
1010  IF M=0 THEN Q=1: RETURN
1015  K=N: P=1
1020  FOR L=M TO 1 STEP -1
1025  P=P*K
1030  IF P>1E34 THEN RETURN
1035  K=K-1
1040  NEXT L
1045  A=P: P=1
1050  FOR J=1 TO M
1055  P=P*J
1060  IF P>1E34 THEN RETURN
1065  NEXT J
1070  Q=A/P
1075  RETURN
```

Der Aufruf

Dem Unterprogramm ab Zeile 1000 werden die Zahlen n und m (m kleiner n) übergeben (Zeile 180 und 200). Es berechnet die Anzahl der Möglichkeiten in Abhängigkeit von der Rechner-Kapazität für die Darstellung von Zahlen. In Zeile 740 wird der berechnete Wert in der Variablen Q ausgegeben.

Die Routine

Der oben genannte Bruch wird im Unterprogramm berechnet. In der Schleife (Zeilen 1020–1040) wird das Zählerprodukt bestimmt und in der Variablen A (Zeile 1045) gespeichert. Die zweite Schleife (Zeilen 1050–1065) ermittelt das Nenner-Produkt P. In Zeile 1070 wird der Quotient Q gebildet. In den Zeilen 1030 und 1060 werden die berechneten Produkte mit der Rechner-Kapazität verglichen. Die Rechnung wird abgebrochen, wenn die Grenze der Zahlendarstellung erreicht wird, um eine Fehlermeldung (overflow error) zu vermeiden. In diesem Fall hat die Variable Q den Wert 0 (Zeile 1005).

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

aktiv computern

```
10 INPUT "WORT";S$
20 FOR I=0 TOπSTEP .01
30 PRINT#7; AT 50*COS
I+120,50*SIN I+80; S$
40 NEXT I
```

Ersetzt man in Zeile 20 STEP .01 durch STEP .1, dann bleiben jeweils Reste des vorherigen Zeichens stehen. Der Grund dafür sind die Leer-Pixelreihen, die jedes Zeichen umranden.

Will man Zahlen ausdrucken, so muß man sie vorher in Zahlenstrings umwandeln:

```
10 FOR P=0 TO 255 STEP 8: LET
P$=STR$ P
20 PRINT#7; AT P,P/8; P$: NEXT P
```

Das Beispiel demonstriert außerdem, daß man auch an denselben Stellen drucken kann, wie mit dem normalen PRINT-Befehl.

Mit der Ausnutzung von Leer-Pixelreihen lassen sich auch 36 Zeichen in einer Bildschirmzeile unterbringen:

```
10 FOR P=0 TO 255-7 STEP 7: LET
P$=(STR$P)(1)
20 PRINT#7; AT P, 100;P$: NEXT P
```

Gordon Detka

```
 1 REM ************************
 2 REM *                      *
 3 REM * PRINT #7;AT x,y;a$   *
 4 REM *                      *
 5 REM ************************
 9 DATA 217
10 DATA 241,241,241,241,241
11 DATA 201,33,171,141,34
12 DATA 36,92,201,98,234
13 DATA 217,122,254,1,40
14 DATA 3,217,207,25,217
15 DATA 231,245,231,254,36
16 DATA 32,9,231,254,43
17 DATA 40,4,254,40,32
18 DATA 2,207,9,241,230
19 DATA 31,246,64,42,75
20 DATA 92,71,126,254,128
21 DATA 40,17,35,94,35
22 DATA 66,184,40,12,43
23 DATA 43,197,205,184,25
24 DATA 193,235,24,234,207
25 DATA 1,35,229,213,217
26 DATA 209,225,217,217,62
27 DATA 0,187,40,166,126
28 DATA 35,27,197,217,193
29 DATA 33,0,91,229,221
30 DATA 225,221,54,0,8
31 DATA 237,91,54,92,254
32 DATA 143,56,6,237,91
33 DATA 123,92,214,143,38
34 DATA 0,111,41,41,41
35 DATA 25,254,32,48,7
36 DATA 17,8,0,237,82
37 DATA 24,1,43,221,117
38 DATA 1,221,116,2,221
39 DATA 102,2,221,110,1
40 DATA 22,0,221,94,0
41 DATA 25,235,197,205,170
42 DATA 34,245,26,221,119
43 DATA 4,241,95,40,10
44 DATA 221,203,4,62,55
45 DATA 203,27,61,24,244
46 DATA 221,86,4,126,163
```

Scan: S. Höltgen

```
160 BORDER 0: PAPER 0:
    BRIGHT 1
: INK 7: CLS
170 LET ya=87: LET xa=127
180 LET a=87: LET b=127
190 FOR x=0 TO 127
200 LET i=x/50.9
210 LET y1=ya-a*SIN (i)
220 LET x2=xa+b*SIN (i)
230 LET y2=x*(175/255)
240 IF x<0 OR x>255 OR
    y1<0 OR
y1>175 THEN GO TO 340
 250 IF x2<0 OR x2>255 OR
    y2<0 O
R y2>175 THEN GO TO 340
260 PLOT 0+x,0+y1
270 PLOT 0+x,175-y1
280 PLOT 255-x,0+y1
290 PLOT 255-x,175-y1
300 PLOT 0+x2,0+y2
310 PLOT 0+x2,175-y2
320 PLOT 255-x2,0+y2
330 PLOT 255-x2,175-y2
340 NEXT x
350 LET a=a-4: LET b=b-6.4
360 IF a<-1 THEN GO TO 390
370 GO TO 190
391 STOP
```

```
UP 160 GRAPHICS 8+16:COLOR 1
BL 170 YA=96:XA=159
SH 180 A=96:B=159
MV 190 FOR X1=0 TO 159
HR 200 I=X1/50.9
LQ 210 Y1=YA-A*SIN(I)
LC 220 X2=XA+B*SIN(I)
FY 230 Y2=X1*(200/335)
JI 240 IF X1<0 OR X1>319 OR
       Y1<0 OR Y1>19
   0 THEN 340
MU 250 IF X2<0 OR X2>319 OR
       Y2<0 OR Y2>19
   0 THEN 340
PK 260 PLOT 0+X1,0+Y1
TV 270 PLOT 0+X1,190-Y1
VP 280 PLOT 319-X1,0+Y1
II 290 PLOT 319-X1,190-Y1
QE 300 PLOT 0+X2,0+Y2
UR 310 PLOT 0+X2,190-Y2
WN 320 PLOT 319-X2,0+Y2
JI 330 PLOT 319-X2,190-Y2
GO 340 NEXT X1
JN 350 A=A-4:B=B-6.4
KJ 360 IF A<-1 THEN 390
PR 370 GOTO 190
QP 390 GOTO 390
```

```
160 CLS
170 ya=199:xa=319
180 a=199:b=319
190 FOR x1=0 TO 319
200 i=x1/101.8
210 y1=ya-a*SIN(i)
220 x2=xa+b*SIN(i)
230 y2=x1*(400/640)
240 IF x1<0 OR x1>639 OR
    y1<0 OR y1>399
THEN 340
```


Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 6)

11. Addition von vielstelligen natürlichen Zahlen

Die nachfolgende Routine addiert natürliche Zahlen M, N mit maximal 254 Stellen und gibt auch eine natürliche Summe aus.

```
 100 REM INPUT
 120 PRINT CHR$(147)
 140 INPUT " ZAHL M ";M$
 160 INPUT " ZAHL N ";N$
 200 M=LEN(M$9: N=LEN(N$)
 220 IF M>254 OR N>254 GOTO 120
 500 GOSUB 1000
 700 REM OUTPUT
 720 PRINT: PRINT" "M$
 740 PRINT" +"N$
 760 PRINT" ";: FOR I=1 TO L+1
 780 PRINT CHR$(192);: NEXT
 800 PRINT:
 820 I=L+2-LEN(A$)
 840 PRINT TAB(I) A$
 860 PRINT: GOTO 140
1000 REM ROUTINE: ADDITION
1005 M=LEN(M$): N=LEN(N$)
1010 L=0: K=M-N
1015 C$=" ": A$=" ": D=0
1020 IF K=0 GOTO 1045
1025 FOR R=1 TO ABS(K)
1030 C$=C$+"0": NEXT
1035 IF K>0 THEN N$=C$+N$
1040 IF K>0 THEN N$=C$+N$
1045 L=LEN(M$)
1050 FOR R=L TO 1 STEP-1
1055 M=VAL(MID$(M$,R,1))
1060 N=VAL(MID$(N$,R,1))
1065 A=M+N+D
1070 IF A>9 THEN D=1
1075 IF A<10 THEN D=0
1080 IF R=1 THEN A$=STR$(A)+A$: GOTO 1090
1085 A$=RIGHT$(STR$(A),1)+A$
1090 NEXT
1095 RETURN
```

Der Aufruf

Dem Unterprogramm ab Zeile 1000 werden die Zahlen als Stringvariable M$, N$ übergeben.

Die Routine

In der Schleife 1050 bis 1090 wird Stelle für Stelle die Addition (Zeile 1065) ausgeführt und das Ergebnis in der Variablen A$ gespeichert.

Scan: S. Höltgen

Zum Sammeln

12. Subtraktion von vielstelligen natürlichen Zahlen

Die nachfolgende Routine subtrahiert natürliche Zahlen und gibt eine ganze Zahl als Ergebnis aus.

```
100 REM INPUT
120 PRINT CHR$(147)
140 INPUT " ZAHL M ";M$
160 INPUT " ZAHL N ";N$
200 M=LEN(M$): N=LEN(N$)
220 IF M>254 OR N>254 GOTO 120
500 GOSUB 1000
700 REM OUTPUT
720 PRINT: PRINT" " M$
740 PRINT" −"N$
760 PRINT" ";: FOR I=1 TO L+1
780 PRINT CHR$(192);: NEXT
800 PRINT:
820 I=L+2−LEN(A$)
840 PRINT TAB(I) A$
860 PRINT: GOTO 140
1000 REM ROUTINE: SUBTRAKTION
1005 M=LEN(M$): N=LEN(N$)
1010 L=0: K=M−N: V=0
1015 C$=" ": A$=" ": D=0
1020 IF K=0 THEN 1045
1025 FOR R=1 TO ABS(K)
1030 C$=C$+"0":NEXT
1035 IF K<0 THEN M$=C$+M$
1040 IF K>0 THEN N$=C$+N$
1045 IF M$<N$ THEN V=−1
1050 IF V THEN S$=M$: M$=N$: N$=S$
1055 L=LEN(M$)
1060 FOR R=L TO 1 STEP−1
1065 M=VAL(MID$(M$,R,1))
1070 N=VAL(MID$(N$,R,1))
1075 A=M−N−D
1080 IF A=>0 THEN D=0: GOTO 1090
1085 M=M+10: D=1: A=M−N
1090 A$=RIGHT$(STR$(A),1)+A$
1095 NEXT
1100 IF V THEN S$=M$: M$=N$: N$=S$
1105 IF V THEN A$="−"+A$
1110 RETURN
```

Der Aufruf

Dem Unterprogramm ab Zeile 1000 werden die Zahlen als Stringvariable M$, N$ übergeben (Zeilen 140, 160). Das Unterprogramm berechnet die Differenz in der Variablen A$. Die Ausgabe erfolgt in Zeile 840.

Die Routine

Die Zeilen 1005 bis 1040 erzeugen Stringvariablen gleicher Länge durch Voranstellen von Nullen. Die Subtraktion (Zeilen 1075 bzw. 1085) erfolgt in der Schleife von 1060 bis 1095 für jede einzelne Stelle (von rechts beginnend). Das Ergebnis wird in der Variablen A$ (Zeile 1080) gespeichert. Die Zeilen 1045, 1050 vertauschen die eingegebenen Zahlen, wenn die erste eingegebene Zahl kleiner ist. Nach der Subtraktion erfolgen der Rücktausch in Zeile 1100 und das Setzen des Minuszeichens in Zeile 1105 für das Ergebnis.

```
250 IF x2<0 OR x2>639 OR
    y2<0 OR y2>399
THEN 340
260 PLOT 0+x1,0+y1
270 PLOT 0+x1,399-y1
280 PLOT 639-x1,0+y1
290 PLOT 639-x1,399-y1
300 PLOT 0+x2,0+y2
310 PLOT 0+x2,399-y2
320 PLOT 639-x2,0+y2
330 PLOT 639-x2,399-y2
340 NEXT x1
350 a=a-13:B=B-20.8
360 IF a<-1 THEN 390
370 GOTO 190
390 GOTO 390
```

```
210 Y1=YA−A∗SIN(I/2)∗
SIN(I/4)
220 X2=XA+B∗COS(I/2)∗(I/2)∗
SIN(I/4)∗TAN(I/2)
```

```
165 BLOCK0,0,319,199,1
210 Y1=YA−A∗SIN(I)∗
(I−π)↑−1
220 X2=XA+B∗COS(I)∗
(1−π)↑−1
```

Flying Lines

Es zählt nicht nur die Quantität, sondern in der Kürze liegt die Würze. Besonders für Grafikprogramme trifft das zu. Alle hier vorgestellten Programme haben eine Grundidee.

Der Computer errechnete gleichzeitig die Punkte zweier gekrümmter Linien, zeigt sie jedoch nicht auf dem Monitor, sondern verbindet sie mit einer geraden Linie. Außerdem haben sie alle folgende Eigenschaften. Sie lassen sich beliebig variieren und laufen sehr schnell ab (ca. 30 Sekunden pro Motiv). Dabei ist die optische Wirkung auf dem Sichtgerät wirkungsvoller als auf dem Drucker.

Programmbeschreibung zu Grafik 1

Es werden zwei Funktionen (FNA und FNB) errechnet und Punkt für Punkt mit einer Geraden verbunden. In den Zeilen 100 und 110 findet eine Koordinatentransformation statt. Die X-Achse rechnet in diesem Beispiel von XU = −PI bis XO = +PI; die Y-Achse von YU = −1 bis YO = 1 (Zeile 140). Diese Bereiche müssen an das Koordinatensystem des Rechners angeglichen werden. Dazu werden beide Systeme in einer Gleichung erfaßt und als BASIC-Funktion eingebracht. In den Zeilen 120 und 130 werden die Funktionen eingefügt. Zeilen 160−220: Die Funktionen

Fortsetzung auf Seite 45

Scan: S. Höltgen

Anwender aufgefordert, das Code-Wort einzugeben. Ist es richtig, kann er mit dem Programm arbeiten. Stimmt die Eingabe nicht mit dem Code-Wort überein, so wird das Programm zerstört.

Nun könnte man natürlich das Programm einfach auflisten und nachsehen, welches das richtige Code-Wort ist. Das bedeutet also: Man müßte versuchen, das Code-Wort und den Zerstörungsbefehl „unsichtbar“ zu machen, so daß sie beim Auflisten nicht mehr zu sehen sind. In den Bemerkungen zum Profi-Listschutz wurden solche Methoden schon eingeführt. Man hängt diesen Kodierer an das eigentliche Programm an. Bevor man dieses Unterprogramm mit RUN 60000 starten kann, muß man wiederum folgendes machen:

Man setzt nach der Zeile, die verschwinden soll, fünf Pluszeichen (+). In unserem Fall wäre das:

```
20 +++++IF A$="VOLKER
MUECKE" THEN ... (usw.)...
30 +++++POKE 776,1
```

Nun startet man das Hilfsprogramm mit RUN 60000. Bei Abfrage der Zeilennummer geben Sie zuerst „20“ ein. Nachdem „OK-OK“ ausgegeben wurde, starten Sie das Programm wiederum mit RUN 60000 und geben diesmal bei der Abfrage „30“ ein. Damit sind die Befehle nach den Zeilennummern 20 und 30 unsichtbar gemacht worden, so daß auf dem Bildschirm folgendes steht:

```
20
30
```

Obwohl der Text unsichtbar bleibt, läuft Ihr Programm weiterhin korrekt und fehlerfrei. Hier noch einige Zerstörungsbefehle und ihre Auswirkungen:

POKE 776,1: Dieser Befehl bewirkt, daß das ganze Programm mit undefinierbaren Zeichen vollgeschrieben wird.

POKE 777,1: Hier wird kein Befehl mehr ausgeführt. Der Cursor springt jeweils in die Ecke links oben.

POKE 770,0: bewirkt, daß eine READY-Ausgabe unendlich oft erfolgt.

Stößt das Programm nun auf einen solchen Zerstörungsbefehl, besteht keine weitere Möglichkeit mehr, als den Computer auszuschalten.

Um das ausführliche Beispiel von oben noch zu verbessern, bie-

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 7)

13. Große Potenzwerte

```
140   PRINT CHR$(147)
160   INPUT" EXPONENT";E%
180   IF E%<2 THEN 140
200   INPUT" BASIS";B%
500   GOSUB 1000
740   PRINT: PRINT
760   PRINT B% "HOCH" E% "="
780   PRINT: PRINT " "A$
800   PRINT: GOTO 160
1000  REM ROUTINE: POTENZWERT
1005  B$="": D$="": A$=""
1010  A$=STR$(B%)
1015  FOR I=1 TO E%-1
1020  D%=0
1025  L=LEN(A$)
1030  FOR N=L TO 1 STEP -1
1035  X%=VAL(MID$(A$(N,1))
1040  X%=X%*B%+D%
1045  D%=INT(X%/10)
1050  Y$=STR$(X%)
1055  IF N=1 THEN 1070
1060  B$=RIGHT$(Y$,1)+B$
1065  GOTO 1075
1070  D$=Y$+B$: B$=""
1075  NEXT N: P=LEN(D$)-2
1080  Q=VAL(LEFT$(D$,2))
1085  IF Q THEN A$=D$: GOTO 1095
1090  A$=RIGHT$(D$,P)
1095  NEXT I: RETURN
```

Der Aufruf

Das Unterprogramm ab Zeile 1000 berechnet für natürliche Zahlen als Exponent E größer als eins zu jeder natürlichen Zahl als Basis B den Potenzwert als (mehrstellige) natürliche Zahl. Dem Unterprogramm müssen die Zahlenwerte E und B übergeben werden. Der Potenzwert wird in der Variablen A$ gespeichert. Die Ausgabe des Wertes erfolgt in Zeile 780.

Die Routine

Die Berechnung erfolgt innerhalb zweier geschachtelter Schleifen. Die äußere Schleife zählt die Anzahl der Multiplikationen E% – 1 (Zeile 1015, Zeile 1095) und beseitigt überflüssige Nullen (Zeilen 1075–1090). In der inneren Schleife (Zeile 1030–1075) werden die Zwischenprodukte mit der Basiszahl B multipliziert. Die Multiplikation erfolgt Stelle für Stelle (Zeile 1040). Das Ergebnis der Multiplikation wird in Zeile 1060 zusammengestellt. Zeile 1045 ermittelt den Übertrag für die nächste Stelle.

Scan: S. Höltgen

14. Multiplikation großer natürlicher Zahlen

```
120   PRINT CHR$(147)
140   INPUT "FAKTOR"; U$
160   INPUT "FAKTOR"; V$
200   U=LEN(M$): V=LEN(N$)
220   IF U+V>254 GOTO 120
500   GOSUB 1000
720   PRINT: PRINT" "U$
740   PRINT" *"V$
760   PRINT" ";: FOR I=1 TO L+1
780   PRINT CHR$(192);: NEXT
800   PRINT:
820   I=L+2-LEN(A$)
840   PRINT TAB(I) A$
860   PRINT: GOTO 140
1000  REM ROUTINE: MULTIPLIKATION
1005  M$="": N$="": D=0
1010  S=LEN(U$): T=LEN(V$)
1015  FOR I=1 TO T
1020  V=VAL(MID$(V$,I,1))
1025  FOR J=S TO 1 STEP-1
1030  U=VAL(MID$(U$,J,1))
1035  U=U*V+D: D=INT(U/10)
1040  Y$=STR$(U)
1045  IF J=1 THEN 1060
1050  M$=RIGHT$(Y$,1)+M$
1055  GOTO 1065
1060  M$=Y$+M$
1065  NEXT J
1070  GOSUB 2000
1075  N$=A$+"0": M$=""
1080  R=LEN(N$)-2
1085  Q=VAL(LEFT$(N$,2))
1090  IF Q=0 THEN N$=RIGHT$(N$,R)
1095  NEXT I: RETURN
```

Der Aufruf

Das Unterprogramm ab Zeile 1000 berechnet das Produkt zweier natürlicher Zahlen als natürliche Zahl, wenn der Produktwert nicht mehr als 250 Stellen hat. Dem Unterprogramm werden die beiden natürlichen Zahlen in den Variablen U$, V$ übergeben. Im Unterprogramm erfolgt ein Sprung in ein weiteres Unterprogramm (Zeile 1070 – GOSUB 2000). Dieses Unterprogramm wurde in Heft 12/85 vorgestellt. Die Berechnung des Produktwertes erfolgt in der Variablen A$.

Die Routine

Die Berechnung des Produktes erfolgt innerhalb zweier geschachtelter Schleifen und mit dem Unterprogramm ab Zeile 2000. Die äußere Schleife (Zeile 1015 bis Zeile 1095) ermittelt die Ziffer (Zeile 1020), mit der der Faktor U$ multipliziert wird. In Zeile 1070 erfolgt die Addition des Zwischenergebnisses (GOSUB 2000). Zeile 1075 verschiebt durch Anhängen einer Null das Zwischenergebnis um eine Stelle nach rechts. Die Zeilen 1080–1090 beseitigen überflüssige Nullen. In der inneren Schleife (Zeilen 1025 bis 1965) erfolgt die Multiplikation der jeweiligen Stelle mit dem Stellenwert V und dem Faktor U$, Stelle für Stelle (Zeile 1035), und das Zusammensetzen des Zwischenergebnisses M$ wird in Zeile 1050 durchgeführt.

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

tet sich folgendes an: Stellen Sie die Code-Wort-Abfrage sowie den Zerstörungsbefehl nicht gleich an den Anfang Ihres Programms. Man könnte so einfach die beiden Zeilen löschen. Verstecken Sie sie lieber in die Mitte Ihres Programms. Nun könnte man natürlich einfach die beiden geschützten Zeilen im Programm suchen und sie wiederum löschen. Darum sollten Sie auf jeden Fall einige wichtige Programmzeilen, ohne die das Programm nicht korrekt läuft, auch schützen, so daß diese als Täuschung für den Knacker dienen.

Hat man nach einiger Zeit sein eigenes Code-Wort vergessen, so besteht natürlich auch die Möglichkeit, dieses Code-Wort zu finden, indem man die unsichtbaren Zeilen eben wieder sichtbar macht. Dies geschieht mit dem Dekodierer des Profi-Listschutzes, welcher wiederum an das eigentliche Programm angehängt wird. Nach RUN 6000 meldet sich das Programm mit „FERTIG!". Somit ist Ihr Programm wieder dekodiert. Selbstverständlich sollte man jeweils danach den Kodierer beziehungsweise Dekodierer wieder aus dem Hauptprogramm nehmen, damit niemand dies dazu ausnutzen könnte, das Programm zu „knakken". *Volker Mücke*

Atari-Grafik unter der Lupe

Wer gut im Bilde sein will, muß den Bildschirmspeicher seines Computers kennen. Monitor oder Fersehgerät haben keinen Speicher, um die Daten festzuhalten, die das Bild erzeugen. Sie müssen laufend mit entsprechenden Signalen versorgt werden, und diese Aufgabe muß der Computer übernehmen.

Atari hat die Kommunikation zwischen Rechner und Bildschirm besonders kreativ gelöst. Ein spezieller Mikroprozessor, ANTIC, holt die Bilddaten aus dem Speicher und übergibt sie an den Video-Chip GTIA, der die digitalen Informationen aus dem Rechner in Signale umwandelt, die von der Bildröhre verarbeitet werden können. Als Mikroprozessor verfügt ANTIC über einen Befehlssatz und Programme. Diese Programme ermöglichen wechselnde Formen der Darstellung auf dem Bildschirm, also verschiedene Grafik-Betriebsarten. In BASIC werden sie mit den Befehlen GRAPHICS 0 bis 15 (mit Textfenster) und 16 bis 31 (ohne Textfenster) aufgerufen. Die Grafik-

Scan: S. Höltgen

```
80 NEXT E,D
90 F=F+1
100 PRINTC(F)
110 GOTO 90
```

Überraschung
Wolfgang Eilhs, 2948 Schortiens 1

```
10 INPUT"Wechselkurs";K
20 INPUT"Ihr Betrag";B
30 R=B*K
40 PRINT"=";R;"DM"
50 END
```

Spirale
Manfred Dellemann, A-6401 Inzing

```
10 A=1
20 S=1
30 FOR N=1 TO 64
40 PRINT" ";N" ";A
50 A=A*2
60 S=S+A
70 NEXT N
80 END
```

Textverarbeitung
Peter Weiland, 3109 Wietze

```
10 INPUT "stick 0,1 oder 2 ";A
20 PRINT "stick";A;
STICK(A);" strig";A;
STRIG(A):GOTO 20
```

Sortiermaschine
Udo Offermann, 5353 Mechrnick

```
10 INPUT A,B,C
15 D=B^2-4*A*C
20 IF D<0 THEN PRINT"k":END
25 Q=-B
30 S=2*A
35 W=SQR(D)
40 X=(Q+W)/S
45 Y=(Q-W)/S
50 IF X=Y THEN PRINTX:END
55 PRINTX,Y
```

Währung
Ernst Olaf, A-8600 Brück/Mür

```
100 CLS:A=RND(-TIME)
110 B=INT(RND(1)*13+1)
120 C=INT(RND(1)*255+1)
130 D=INT(RND(1)*920+1)
140 SOUND B,C:VPOKE D,C:
GOTO 110
```

Reiskornproblem
Ernst Olaf, A-8600 Brück/Mür

```
1 SCREEN2
2 FOR X=10 TO 255 STEP 2
3 Y=100-(70*SIN(2*3*(X-10)/110))
4 CIRCLE(X,Y),20,1,1.5
5 NEXT
6 GOTO 6
10 REM
```


Programm-bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 8)

15. Intervallschachtelung von Pi nach Pythagoras

```
100 REM INPUT
120 PRINT CHR$(147)
140 INPUT" RADIUS";R
160 INPUT" TEILUNGEN";T
180 PRINT
500 GOSUB 1000
700 REM OUTPUT
720 PRINT TAB(4)"UMFANG DES"
740 PRINT " INNEN-"N"-ECKS:"
750 PRINT TAB(3) UI
760 PRINT TAB(4)"UMFANG DES"
780 PRINT " AUSSEN-"N"-ECKS:"
790 PRINT TAB(3) UA
800 PRINT: PRINT PI"<π<" PA
820 PRINT: GOTO 140
1000 REM ROUTINE (PYTHAGORAS)
1005 S=1: N=6
1010 W=SQR(1-(S/2)↑2)
1015 SA=S/W
1020 UI=N∗R∗S
1025 UA=N∗R∗SA
1030 IF T=0 THEN 1050
1035 S=SQR(2∗(1-W))
1040 N=2∗N: T=T-1
1045 GOTO 1010
1050 PI=UI/R∗.5
1055 PA=UA/R∗.5
1060 RETURN
```

Der Aufruf

Die Unter-Routinen ab Zeile 1000 berechnen den Umfang von regelmäßigen Vielecken, die im oder am Kreis liegen. Es ist ein regelmäßiges Sechseck vorgegeben. Der Routine wird der Radius R des Kreises und die gewünschte Zahl der Teilungen T des vorgegebenen Sechsecks übergeben. Das Unter-Programm berechnet den Umfang des regelmäßigen Innenecks in der Variablen UI, den Umfang des regelmäßigen Außenecks in der Variablen UA und die daraus resultierenden Näherungswerte für die Zahl Pi.

Die Routine

Diese Routine verwendet den Satz des Pythagoras zur Berechnung der genannten Werte. In Zeilen 1005 bis 1025 werden die Ausgangswerte des regelmäßigen

Scan: S. Höltgen

Sechsecks berechnet. In Zeile 1040 wird die Teilervariable T erniedrigt und dementsprechend die Eckenzahl N verdoppelt. Ist die Anzahl der gewünschten Teilungen ausgeführt (Zeile 1030, T = 0), erfolgen die Berechnung der Näherungswerte für die Variablen PI und PA in Zeile 1050 bzw. 1055 und der Rücksprung. Der Nachteil dieser Routine besteht darin, daß bei einer Teilerzahl von 14 die Rechengenauigkeit der meisten Homecomputer nicht mehr ausreicht und deshalb falsche Werte ausgegeben werden.

16. Intervallschachtelung von Pi nach Gregor

```
 100 REM INPUT
 120 PRINT CHR$(147)
 140 INPUT" RADIUS";R
 160 INPUT" TEILUNGEN";T
 180 PRINT
 500 GOSUB 1000
 700 REM OUTPUT
 720 PRINT TAB(4)"UMFANG DES"
 740 PRINT " INNEN-"N"-ECKS:"
 750 PRINT TAB(3) UI
 760 PRINT TAB(4)"UMFANG DES"
 780 PRINT " AUSSEN-"N"-ECKS:"
 790 PRINT TAB(3) UA
 800 PRINT: PRINT PI"<π<" PA
 820 PRINT: GOTO 140
1000 REM ROUTINE (GREGORY)
1005 UI=6*R: UA=4*R*SQR(3)
1010 N=6
1015 IF T=0 THEN 1040
1020 N=2*N: T=T-1
1025 UA=2*UA*UI/(UI+UA)
1030 UI=SQR(UI*UA)
1035 GOTO 1015
1040 PI=UI/R*.5
1045 PA=UA/R*.5
1050 RETURN
```

Der Aufruf

Die Unter-Routinen ab Zeile 1000 berechnen den Umfang von regelmäßigen Vielecken, die im oder am Kreis liegen. Es ist ein regelmäßiges Sechseck vorgegeben. Der Routine wird der Radius R des Kreises und die gewünschte Zahl der Teilungen T des vorgegebenen Sechsecks übergeben. Das Unter-Programm berechnet den Umfang des regelmäßigen Innenecks in der Variablen UI, den Umfang des regelmäßigen Außenecks in der Variablen UA und die daraus resultierenden Näherungswerte für die Zahl Pi in den Variablen PI und PA.

Die Routine

Diese Routine verwendet ein Verfahren, das von dem Mathematiker James Gregory angegeben worden ist. Allerdings gibt es auch hier eine Grenze bei T = 14 aufgrund der Rechengenauigkeit des Rechners. Zeilen 1005 und 1010 berechnen die Anfangswerte des regelmäßigen In- und Außensechsecks. In Zeile 1015 erfolgt der Rücksprung, wenn die Anzahl der gewünschten Teilungen ausgeführt ist (T = 0), über die Zeilen 1040 und 1045, in denen die Näherungswerte für PI und PA berechnet werden.

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

Atari

Schrift
Alexander Dub, A-1210 Wien

```
10 FOR I=39974 TO 39996
STEP 2:POKE I,5:NEXT I
20 POKE 709,14
30 POKE 710,14
40 POKE 622,255
50 ? "ATARI"
60 GOTO 50
```

Farbgrafik
Harald Weinreich, 2102 Hamburg 93

```
1 GRAPHICS 9:TRAP 4
2 I=I+1:COLOR I*I/298:
PLOT 0,I:DRAWTO 79,I
3 GOTO 2
4 POKE 53274,184:POKE 53274,
64:GOTO 4
```

Die Zahl „PI“
Marek Hahn, 5357 Swisstal-Odendorf

```
1 FOR A=1 TO 23 STEP 2
2 D=(1/A)*((1/2^A)+(1/3^A))
3 C=C+D
4 C=-C
5 ? ABS(C*4)
6 NEXT A
```

Diskette voll?
Matthias Heigl, 8481 Altenstadt/Wn.

```
32762 REM FREE SPACE ON DISC
32763 CLOSE #1:CLR
32764 DIM A$(17)
32765 OPEN #1,6,0,"D:.FREE"
32766 INPUT #1;A$:? A$;"
ON DISC"
32767 END
```

Strichgrafik
Metehan Sen, 4500 Osnabrück

```
1 GRAPHICS 24
2 COLOR 1
3 FOR A=0 TO 320 STEP 15.9
4 FOR X=0 TO 320 STEP 53.1
5 PLOT A,190:DRAWTO X,95
6 PLOT A,0:DRAWTO X,95
7 NEXT X
8 NEXT A
9 GOTO 9
```

Sound mit Joystick
Karim Saleh, 3033 Schwarmstedt

```
10 A=765
20 INPUT B
30 S=STICK(0)
40 IF S<12 AND A>0 THEN A=A-1
45 IF S<8 THEN A=A+2
50 SOUND 0,A,B,10
60 ? A
70 GOTO 30
```

Rückwärts
Udo Saatmann, 5144 Wegberg

```
10 DIM A$(50)
20 ? "VORW!"
30 INPUT A$
```

Scan: S. Höltgen

1030 bis 1100: Ebenso wie den statistischen Mittelwert kann man vom Grafikbildschirm aus auch ein Vergleichsdiagramm zeichnen lassen. Hierzu werden die Werte für die Bildschirmpunkte um vier gegenüber denen des Hauptdiagramms reduziert, so daß die Vergleichsstatistik jeweils die Balken der Hauptstatistik überlagert. Als zweites Unterscheidungsmerkmal werden die Flächen der Vergleichsstatistik nicht ausgefüllt. Theoretisch ließen sich noch weitere Vergleichsstatistiken auf diese Weise übereinanderlagern. Aufgrund der mangelnden Grafik- bzw. Farbgrafikmöglichkeiten (in der höchsten Auflösungsstufe lassen sich meist nur zwei Farben darstellen) würde dies jedoch die ,,klare Sicht" mindern.

1120 und 1130: Diese Unterprogramme gelten nur für den Schneider. Hier wird die Bildschirmfarbe so verändert, daß die Textzeichen invertiert bzw. normal erscheinen. Besitzer anderer Computer können – sofern vorhanden – entsprechende Befehle einsetzen oder lediglich die Buchstabenfarbe für die Menüeinblendungen verändern.

1140 und 1150: Da sowohl beim Abspeichern als auch zum Einlesen von Daten nach dem Dateinamen und der Entscheidung, ob die Daten einer Haupt- oder Vergleichsstatistik zuzuordnen sind, gefragt wird, ist es speichersparend, hierfür ein kurzes Unterprogramm einzufügen, das von den jeweiligen Auswahlpunkten im Hauptmenü aus angesprungen wird. *Alfred Görgens*

52 MSX-Tricks

Bei den MSX-Rechnern enthüllen sich bei längerem Studium eine Fülle von Kniffen, die für eigene Programmiervorhaben eine wertvolle Hilfe sein können.

Manipulationen des Bildschirms

Bildschirm ausschalten
Der Bildschirm läßt sich mit DEFUSR&H41:X%=USR (1) ausschalten. Jetzt kann die Ausgabe wie gewohnt auf den Bildschirm erfolgen, es ist jedoch nichts zu sehen. Der Rechner ist jedoch schneller, da nicht immer der Interrupt des Videoprozessors abgewartet werden muß.

Bildschirm einschalten
DEFUSR=&H44:X%=USR (1)

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 9)

17. Wertetabellen von rationalen Funktionen

Die nachfolgende Routine berechnet für eine rationale Funktion (z.B. $y = 2x^3 + 4x^2 - 2x + 3$) und deren erste und zweite Ableitungsfunktion die Funktionswerte (Y-Werte), wenn der X-Wert vorgegeben wird. Der Grad einer rationalen Funktion ist die größte Hochzahl, die bei den Potenzen der Variablen X auftritt.

```
100 REM INPUT
120 PRINT CHR$ (147)
140 F=0 : G=0
200 OPEN1,0
210 IF F THEN 300
220 PRINT" GRAD ";
230 INPUT#1,G
240 FOR I=G TO 0 STEP -1
250 PRINT
260 PRINT" ZAHL VOR X↑"I
270 PRINT" ";
280 INPUT#1,K(I)
290 NEXT
300 PRINT
310 PRINT" X-WERT "
320 PRINT" ";
330 INPUT#1,X
340 CLOSE1
500 GOSUB 1000
700 REM OUTPUT
720 PRINT
740 PRINT" Y="Y
760 PRINT" Y1="Y1
780 PRINT" Y2="Y2
800 PRINT" NEUE FUNKTION? J!"
810 GET T$ : IF T$="" THEN 810
820 IF T$="J" THEN 100
830 F=1 : GOTO 200
1000 REM ROUTINE: WERTETABELLE
1005 Y=0 : Y1=0 : Y2=0
1010 FOR J=G TO 0 STEP -1
1015 Y=Y+K(J)*X↑J
1020 Y1=Y1+J*K(J)*X↑(J-1)
1025 Y2=Y2+J*(J-1)*K(J)*X↑(J-2)
1030 NEXT
1035 RETURN
```

Der Aufruf

Dem Unterprogramm müssen ab Zeile 1000 die Faktoren K(I), die vor den X-Potenzen stehen, und der Grad der rationalen Funktion in der Variablen G übergeben werden (Zeile 230, Zeile 280).

Die Routine

Die Funktionswerte werden in einer Schleife berechnet (Zeilen 1010 bis 1030). Zeile 1015 berechnet den Y-Wert, Zeile 1020 den Y1-Wert (erste Ableitungsfunktion), Zeile 1025 den Y2-Wert (zweite Ableitungsfunktion). Diese Werte Y, Y1, Y2 werden in den Zeilen 740, 760, 780 ausgegeben.

Scan: S. Höltgen

18. Nullstellen von rationalen Funktionen

Diese Routine löst Gleichungen mit einer Variablen beliebigen Grades bzw. berechnet die Nullstellen von rationalen Funktionen.

```
100 REM INPUT
120 PRINT CHR$ (147)
140 F=0 : G=0 : NF=0
200 OPEN1,0
210 IF F THEN 300
220 PRINT" GRAD ";
230 INPUT#1,G
240 FOR I=G TO 0 STEP -1
250 PRINT
260 PRINT" ZAHL VOR X↑"I
270 PRINT" ";
280 INPUT#1,K(I)
290 NEXT
300 PRINT
310 PRINT" XA-WERT "
320 PRINT" ";
330 INPUT#1,XA
340 PRINT
350 PRINT" XE-WERT "
360 PRINT" ";
370 INPUT#1,XE
380 CLOSE1
500 GOSUB 1000
700 REM OUTPUT
720 PRINT : IF NF THEN 800
740 PRINT" NULLSTELLE"
760 PRINT
780 PRINT" ("X"/"Y")"
790 PRINT
800 PRINT" NEUE FUNKTION? J!"
810 GET T$ : IF T$="" THEN 810
820 IF T$="J" THEN 100
830 F=1 : NF=0 : GOTO 200
1000 REM ROUTINE
1005 FOR L=1 TO 10 : N=.1↑L
1010 IF N=.1 THEN 1020
1015 IF N<.1 THEN 1030
1020 FOR X=XA TO XE+N STEP N
1025 GOTO 1035
1030 FOR X=Z TO Z+.1 STEP N
1035 Z=X-N : Y=0 : W=0
1040 FOR J=G TO 0 STEP -1
1045 Y=Y+K(J)*X↑J
1050 W=W+K(J)*Z↑J
1055 NEXT
1060 A=SGN(Y)+SGN(W)
1065 IF A=0 THEN 1095
1070 IF X>XE THEN 1100
1075 IF ABS(Y)<1.E-8 THEN Y=0
: RETURN
1080 IF W=Y THEN RETURN
1085 PRINT X;Y
1090 NEXT X
1095 NEXT L
1100 NF=1 : RETURN
```

Der Aufruf

Dem Unterprogramm ab Zeile 1000 müssen die Faktoren K(I), die vor den X-Potenzen stehen, der Grad (das ist die größte Hochzahl) in der Variablen G und der Anfangswert XA sowie der Endwert XE (das ist der vermutete Bereich der gesuchten Nullstelle) übergeben werden. Die Routine sucht den eingegebenen Bereich in Zehntel-Schritten ab. Wird eine Nullstelle (Lösung) vermutet, so wird diese Stelle durch Intervallschachtelung bestimmt.

Die Routine

Die äußere Schleife (Zeilen 1005 und 1095) legt die Intervallänge N bei einer Schachtelung fest. Wird keine Nullstelle gefunden, so bleibt N = 0.1 (Zeile 1010) für den zu untersuchenden Bereich bestehen. Wird eine Nullstelle (Lösung) vermutet (Zeile 1060 bzw. 1065), dann wird die Intervallschachtelung vorgenommen. Die Werte der gefundenen Nullstelle werden den Variablen X, Y übergeben und in Zeile 780 ausgegeben. Zeile 1085 gibt den jeweils untersuchten X-Wert und den dazugehörigen Y-Wert auf dem Bildschirm aus (Wertetabelle). Wenn diese Ausgabe nicht gewünscht wird, kann die Zeile durch 1085 REM ersetzt werden.

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

schaltet den Bildschirm wieder ein. Alle während der Ausschaltphase auf den Bildschirm ausgegebenen Werte werden sichtbar.

Sprites neu initialisieren
Um alle auf dem Bildschirm befindlichen Sprites zu löschen und die Werte auf ihren Urzustand zu bringen, muß mit dem SCREEN-Befehl und der Angabe der Spritegröße operiert werden. Dabei wird jedoch auch der komplette Bildschirm gelöscht.
DEFUSR=&H69:X%=USR (1)
löscht alle Sprites in ihren Formen, Farben und Positionen, ohne daß dabei der Bildschirmhintergrund gelöscht wird.

Cursorfarbe verändern
Der Cursor ist, da er die gleiche Farbe wie die ausgegebenen Zeichen hat, meist sehr schwer auf dem Bildschirm zu finden. Im SCREEN 1-Modus kann man jedoch die Farbe mit
+VPOKE BASE (6) +31, 16 * *Vordergrundfarbe* + *Hintergrundfarbe*
den eigenen Wünschen anpassen.

Anzahl der Bildschirmzeilen
Mit
POKE &HF3B1, *Zeilenzahl*
kann man in den Bildschirmmodi SCREEN 0 und SCREEN 1 die Anzahl der Zeilen für den Bildschirm festlegen, wobei *Zeilenzahl* im Bereich 1 bis 24 liegen muß. Gibt man z.B. POKE &HF3B1,10, so erfolgt ab sofort die Ausgabe nur noch in den oberen 10 Bildschirmzeilen, ohne daß dabei der Rest des Bildschirms gelöscht wird. Die Anzahl der Bildschirmzeilen wird auch bei der Umschaltung mit einem SCREEN-Befehl nicht gelöscht.

Zeilenbreite sofort einstellen
Die Zeilenbreite läßt sich ebenfalls auf dem Bildschirm verändern, ohne daß dabei der Bildschirm gelöscht wird, wie dies bei der WIDTH-Anweisung der Fall ist.
POKE &HF3B0, *Zeilenbreite*
stellt die neue Breite der Bildschirmzeile sofort ein, ohne den Bildschirm dabei zu löschen. Der Wert für *Zeilenbreite* muß bei SCREEN 0 im Bereich 1 bis 40 und bei SCREEN 1 im Bereich 1 bis 32 liegen.

Zeilenbreite voreinstellen
Die Zeilenbreite läßt sich auch in ihren Grundeinstellungen verändern, die dann jedesmal bei einer Umschaltung durch einen SCREEN-Befehl automatisch eingestellt wird.
POKE &HF3AE, *Zeilenbreite*
wirkt auf den SCREEN 0-Modus.
POKE &HF3AF, *Zeilenbreite*
wirkt auf den SCREEN 1-Modus.
Die Einstellung auf diese neue Zeilenbreite erfolgt jedoch erst bei der

```
5Q 6050 GOTO 6015
LQ 6100 POSITION 28,12:? "Press START";:C
    LOSE #1:POKE 764,255:POKE 512,204:POKE
     513,6:POKE 54286,192
KW 6150 IF PEEK(KL)<>6 THEN 6150
OY 6200 POSITION 1,12:? LEER$:POSITION 0,
    11:? "      No more files...
         ":FOR W=1 TO 200:NEXT W:GOTO MENUE
IL 6500 REM
BM 6505 REM DELETE
IO 6510 REM
LZ 6520 ITEM$=" DELETE ":HELP$="Are you s
    ure (Y/N) ?               ":GOSUB I
    NSTR
DW 6530 IF PEEK(764)=255 THEN 6530
LG 6540 IF PEEK(764)<>43 THEN GOTO BACKG
KD 6550 W=USR(DEL)
XK 6560 FOR X=0 TO 19 STEP 2:POKE 31744+X
    *256,8:POKE 31744+X*256+255,9:NEXT X
LE 6999 REM
DT 7000 REM BACKGROUND
IG 7001 REM
VM 7005 FLAG=2
VA 7007 FOR X=8 TO 0 STEP -4:SOUND 0,X,10
    ,X:NEXT X:SOUND 0,0,0,0
NR 7008 POKE 88,20:POKE 89,154:POSITION 0
    ,1:? "     Choose character - press but
    ton !  ":POKE 39544,93
XK 7009 POSITION 0,0:? "B=new char.ColorD
    eleteE O L S -=+* I K"
UZ 7010 POKE 54279,120:POKE 559,46:POKE 5
    3277,3:POKE 704,14:POKE HPOSP0,127
FP 7020 POKE 560,115:POKE 561,123:POKE 75
    6,148:POKE 512,204:POKE 513,6
BG 7035 GOSUB 10000
CH 7036 ITEM$="          ":HELP$=" Set choo
    sed character where you want":GOSUB IN
    STR
YL 7039 X=61:POKE 54286,192:POKE 704,104:
    PLACE=19
LY 7040 A=STICK(0)
TX 7041 POKE 704,14
YI 7045 IF PEEK(753)=3 THEN POKE 31233+X,
    0:POKE 31232+X,0:GOTO MENUE+6
OS 7046 IF PEEK(KL)=3 THEN PLACE=PLACE+1:
    POKE KL,0:GOTO 7050
TA 7047 IF PEEK(KL)=5 THEN PLACE=PLACE-1:
    POKE KL,0:GOTO 7050
UF 7048 IF PEEK(KL)=6 THEN PLACE=19:POKE
    KL,0
EN 7050 IF PLACE<0 THEN PLACE=39:GOTO 705
    2
PF 7052 IF PLACE>39 THEN PLACE=0
QW 7053 POKE HPOSP0,47+PLACE*4
NX 7054 POKE 704,0
YH 7055 IF A=15 THEN 7080
NV 7060 IF A=14 OR A=10 OR A=6 THEN X=X-4
    :GOTO 7070
TH 7065 IF A=13 OR A=9 OR A=5 THEN X=X+4
JE 7070 W=USR(CSCROL,A)
IQ 7071 IF X>113 THEN POKE 31345,0:POKE 3
    1346,0:X=37:GOTO 7073
AT 7072 IF X<37 THEN POKE 31270,0:POKE 31
    269,0:GOTO 7080
OS 7073 POKE 31233+X,252:POKE 31229+X,0:P
    OKE 31237+X,0:POKE KL,0
MM 7074 POKE 31232+X,132:POKE 31228+X,0:P
    OKE 31236+X,0
KF 7080 IF STRIG(0)=1 THEN 7040
TV 7300 SOUND 0,10,10,14:SOUND 0,0,0,0
WB 7398 ORT=31744+PLACE+PEEK(31618)+(X-37
    )*64
YK 7399 IF PEEK(ORT)<>0 THEN POKE ORT,0:G
    OTO 7040
QM 7400 SOUND 0,255,10,14:SOUND 0,0,0,0:P
```


Programm-bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 10)

19. Bestimmtes Integral von rationalen Funktionen

```
100 REM INPUT
120 PRINT CHR$ (147)
140 F=0 : G=0
200 OPEN1,0
210 IF F THEN 300
220 PRINT" GRAD ";
230 INPUT#1,G
240 FOR L=G TO 0 STEP -1
250 PRINT
260 PRINT" ZAHL VOR X↑"L
270 PRINT" ";
280 INPUT#1,K(L)
290 NEXT
300 PRINT
310 PRINT" OBERE GRENZE "
320 PRINT" ";
330 INPUT#1,XO
340 PRINT
350 PRINT" UNTERE GRENZE "
360 PRINT" ";
370 INPUT#1,XU
380 CLOSE1
500 GOSUB 1000
700 REM OUTPUT
720 PRINT
740 PRINT" O ="O
760 PRINT" U ="U
780 PRINT" FLAECHE="I
800 PRINT" NEUE FUNKTION ? J!"
810 GET T$ : IF T$="" THEN 810
820 IF T$="J" THEN 100
830 F=1 : GOTO 200
1000 REM ROUTINE : INTEGRAL
1005 O=0 : U=0 : I=0
1010 FOR J=G TO 0 STEP -1
1015 L=J+1 : K=K(J)/L
1020 O=O+K*XO↑L
1025 U=U+K*XU↑L
1030 NEXT
1035 I=O-U
1040 RETURN
```

Mit dem bestimmten Integral wird der Flächeninhalt zwischen der x-Achse und dem Graph zwischen der oberen und unteren Grenze berechnet. Verläuft der Graph der Funktion unterhalb der x-Achse, ist der Flächeninhalt negativ. Nullstellen, die zwischen der oberen und unteren Grenze liegen, müssen vorher ermittelt (vgl. HC März 85) und dann das bestimmte Integral entsprechend teilweise berechnet werden.

Scan: S. Höltgen

Der Aufruf

Dem Unterprogramm müssen ab Zeile 1000 die Faktoren K(L), die vor den x-Potenzen stehen, der Grad G der Funktion sowie die untere und obere Grenze XU bzw. XO übergeben werden (Zeilen 230, 280, 330, 370).

Die Routine

Die Werte für die obere und untere Grenze werden in der Schleife Zeile 1010 bis Zeile 1030 berechnet. In Zeile 1035 wird der Flächeninhalt I berechnet, der in der Zeile 780 ausgegeben wird.

20. Extrempunkte von rationalen Funktionen

```
1000 REM ROUTINE : EXTREMPUNKTE
1005 FOR L=1 TO 10 : N=.1↑L
1010 IF N=.1 THEN 1020
1015 IF N<.1 THEN 1030
1020 FOR X=XA TO XE+N STEP N
1025 GOTO 1035
1030 FOR X=Z TO Z+.2 STEP N
1035 Z=X-N : Y=0 : W=0
1040 FOR J=G TO 0 STEP -1
1045 Y=Y+J * K(J) * X↑(J-1)
1050 W=W+J * K(J) * Z↑(J-1)
1055 NEXT
1060 A=SGN(Y)+SGN(W)
1065 IF A=0 THEN 1090
1070 IF X>XE THEN NF=1 : RETURN
1075 IF ABS(Y)<1.E-8 THEN 1095
1080 IF W=Y THEN 1095
1085 NEXT X
1090 NEXT L
1095 FOR J=G TO 0 STEP -1
1100 Y=Y+K(J) * X↑J : NEXT
1105 RETURN
```

Die nachfolgende Routine berechnet die x- und y-Koordinate eines relativen Extrempunktes (rel. Hoch- bzw. Tiefpunkte) für rationale Funktionen. Die Input- und Output-Routine wurde weggelassen. Zum Testen des Unterprogramms können die Ein- und Ausgabe-Routinen für die Berechnung der Nullstellen von rationalen Funktionen aus dem März-HC-Heft übernommen werden. In Zeile 740 ist lediglich das Wort „Nullstelle" durch das Wort „Extremum" zu ersetzen.

Der Aufruf

Dem Unterprogramm werden ab Zeile 1000 der Grad G der Funktion (Zeile 230), die Koeffizienten K(I) (Zeile 280) und der Suchbereich mit dem Anfangswert XA und dem Endwert XE übergeben (Zeile 310; Zeile 350).

Die Routine

Innerhalb zweier Schleifen (Schleife 1: Zeile 1005–1090, Schleife 2: Zeile 1020 bzw. 1030 und Zeile 1085) wird der x-Wert berechnet. In der dritten Schleife (Zeile 1095 bzw. 1100) wird die y-Koordinate ermittelt. Wird kein Extremum innerhalb der angegebenen x-Werte XA bzw. XE gefunden, so wird in Zeile 1070 das Flag NF gesetzt. Die Ausgabe der x- bzw. y-Koordinate eines Extremwertes erfolgt in Zeile 780 der Output-Routine (HC-Heft, März 85).

```
   OKE ORT,CURR:GOTO 7040
LF 7999 REM
SZ 8000 REM SCROLLING
IH 8001 REM
AL 8060 FOR X=0 TO 7:SAV=PEEK(S+X):IF SAV
   >127 THEN POKE S+X,(PEEK(S+X)-128)*2+1
   :NEXT X:GOTO 8200
RU 8061 POKE S+X,PEEK(S+X)*2:NEXT X:GOTO
   8200
CW 8070 FOR X=0 TO 7:SAV=PEEK(S+X):IF SAV
   /2<>INT(SAV/2) THEN POKE S+X,(PEEK(S+X
   )-1)/2+128:NEXT X:GOTO 8200
WN 8071 POKE S+X,PEEK(S+X)/2:NEXT X:GOTO
   8200
BR 8140 SAV=PEEK(S):FOR I=1 TO 7:POKE S+I
   -1,PEEK(S+I):NEXT I:POKE S+7,SAV:GOTO
   8200
MY 8150 SAV=PEEK(S+7):FOR I=7 TO 0 STEP -
   1:POKE S+I,PEEK(S+I-1):NEXT I:POKE S,S
   AV:GOTO 8200
AW 8200 GOSUB 10100
SP 8205 IF FLAG=2 THEN GOTO BACKG
IQ 8210 GOTO EDITPLUS
LG 8999 REM
SJ 9000 REM INVERT
II 9001 REM
SC 9010 ITEM$=" INVERT ":HELP$=LEER$:GOSU
   B INSTR
EH 9020 FOR X=0 TO 7:POKE CH+CURR*8+X,255
   -PEEK(CH+CURR*8+X):NEXT X:A=USR(SCOL,C
   H+CURR*8,40079):IF FLAG=1 THEN 1028
PG 9030 GOTO BACKG
LH 9999 REM
OT 10000 REM SEARCH
BH 10001 REM
QX 10010 POKE 54286,192
CM 10015 POKE 40860,93
UN 10020 A=STICK(0)
CY 10025 IF PEEK(753)=3 THEN GOTO MENUE+6
OA 10030 IF A=7 THEN CO=CO+1:GOTO 10045
DW 10040 IF A=11 THEN CO=CO-1
AM 10045 IF CO<20 THEN CO=147:GOTO 10051
TX 10050 IF CO>147 THEN CO=20
NN 10051 CURR=PEEK(1573):PCURR=CURR
KI 10055 IF STRIG(0)=0 THEN FOR I=14 TO 0
    STEP -1:SOUND 0,6,10,I:NEXT I:SOUND 0
   ,0,0,0:RETURN
CB 10060 POKE 1573,CO-20:POKE 31611,CO-20
   :POKE 31615,CO-20:GOTO 10020
EA 10098 REM
IA 10099 REM EINFARBIGES BITMUSTER
LV 10100 A=USR(SCOL,CH+CURR*8,40079)
DI 10120 RETURN
FG 10999 REM
NZ 11000 REM KILL
BJ 11001 REM
GA 11005 ITEM$="  KILL  ":HELP$=LEER$:GOS
   UB INSTR
KZ 11010 FOR X=0 TO 7:POKE CH+CURR*8+X,0:
   NEXT X:A=USR(SCOL,CH+CURR*8,40079):IF
   FLAG=1 THEN GOTO EDITPLUS
ZY 11020 GOTO BACKG
FY 19999 REM
CL 20000 REM MENUE
BI 20001 REM
FB 20003 ITEM$=" CHOICE ":HELP$="
    Your choice ... ?                 ":GOSUB
   INSTR
JN 20006 T=764
JV 20010 IF PEEK(T)=42 THEN 1000
NH 20020 IF PEEK(T)=18 THEN 2000
IT 20030 IF PEEK(T)=8 THEN 3000
EI 20040 IF PEEK(T)=0 THEN 4000
QV 20045 IF PEEK(T)=14 OR PEEK(T)=15 OR P
   EEK(T)=6 OR PEEK(T)=7 THEN S=CH+CURR*8
```

Scan: S. Höltgen

1. Das erste Zeichen (inverses Pik/CHR$(129), gleichzeitig Steuerzeichen für die Farbe Orange).
2. Der Name ist immer 15 Zeichen lang.
3. Fehlende Zeichen (bis 15) werden mit Freiräumen (Space/CHR$(32)) aufgefüllt.

Da die Eingabe dieses Formates über die Tastatur schwierig ist, übernimmt das Programm diese Arbeit. Beim Laden und Speichern von Koala-Pad-Bildern braucht nur noch der Kennbuchstabe und der Bildname eingegeben werden.
Beispiel: Original Name ,,PIC A ANIMAL"
Eingabe ,,A ANIMAL"

Beispiel

Das Koala-Pad-Bild mit dem Namen ,,PIC A ANIMAL" soll als Print-Shop-Bild abgespeichert werden:
Menüpunkt (1) ,,Grafik laden" anwählen.
Als Programmnamen ,,A ANIMAL" eingeben.
,,KOALA (J/N)" mit ,,J" beantworten.

Nach dem Laden des Bildes Menüpunkt (3) anwählen (,,Grafik verschieben"). Die Frage ,,WELCHE GRAFIK VERSCHIEBEN (1/2/3)?" mit ,,3" beantworten; es soll ja eine Koala-Pad-Grafik verschoben werden (die Grafik befindet sich im Computer ab der Speicherstelle $6000). Die Frage ,,WOHIN SCHIEBEN (1/2/3)?" mit ,,2" beantworten; die Koala-Pad-Grafik soll ja in den Print-Shop-Bereich verschoben werden (Print-Shop-Grafiken stehen im Rechner ab $4000). Nach zirka einer Sekunde meldet der Computer mit ,,OK", daß die Verschiebung abgeschlossen ist (die Grafik steht nun bei $6000 und bei $4000). Nun kann mit (4) die gewünschte Grafik abgespeichert werden. Die Frage ,,WELCHE GRAFIK SPEICHERN (1–5)?" wird mit ,,3" (Print-Shop) beantwortet. Nach der Angabe des neuen Bildnamens wird das Bild als Print-Shop-Grafik abgespeichert und kann nun mit dem Programm Print-Shop verarbeitet werden. Sollte sich auf der Diskette schon ein Programm oder Bild mit gleichem Na-

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 11)

21. Wert von Determinanten

```
100 REM INPUT
110 DIM A(20,20): DIM S(20):
    DIM H(20)
120 PRINT CHR$(147)
130 OPEN 1,0: PRINT
140 PRINT TAB(2)" ANZAHL
    DER REIHEN: ";
150 INPUT#1,N: PRINT
160 PRINT: PRINT TAB(2);
170 FOR I=1 TO N
180 FOR J=1 TO N
190 INPUT#1,A(I,J)
200 NEXT J
210 PRINT: PRINT TAB(2):
    NEXTI
220 CLOSE 1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT: PRINT:
720 PRINT TAB(3)"WERT
    DER DETERMINANTE":
730 PRINT: PRINT TAB(5) D:
    PRINT
740 PRINT TAB(2)" NEUE
    WERTE ?"
750 PRINT TAB(2)"
    TASTE J ODER N !"
760 G$=""
770 GET G$: IF G$="J"
    GOTO 120
780 IF G$="N" THEN END
790 GOTO 760
1000 REM ROUTINE:
     DETERMINANTE
1005 M=N: D=1:
     FOR K=1 TO M
1015 IF A(1,1)=0 THEN GOSUB
     1075
1020 IF D=0 THEN 1070
1025 FOR I=M TO 1 STEP-1
1030 FOR J=2 TO M
1035 A(J,I)=A(J,I)-A(1,I)*
     A(J,1)/A(1,1)
1040 NEXTJ: NEXTI
1045 S(K)=A(1,1): M=M-1
1050 FOR I=1 TO M: FOR J=
     1 TO M
1055 A(I,J)=A(I+1,J+1)
1060 NEXT J: NEXT I: NEXT K
1065 FOR I=1 TO N:D=D*
     S(I): NEXT
1070 RETURN
1075 FOR L=1 TO M
1080 IF A(L,1)<>0 THEN 1090
1085 NEXT: D=0: RETURN
1090 FOR P=1 TO M
1095 H(P)=A(1,P): A(1,P)=
     -A(L,P)
1100 A(L,P)=H(P): NEXT:
     RETURN
```

Der Aufruf

Die Unterroutinen ab Zeile 1000 berechnen den Wert einer symmetrischen n-reihigen Determinante. Dem Unterprogramm werden die Zahlrenreihen der Determinante in den Variablen A(I,J) und die Zahl der Reihe n in der Variablen N(Zeile 150 bzw. 190) übergeben. Die Routine berechnet den Wert der Determinante in der Variablen D. Dieser Wert wird in Zeile 730 ausgegeben. Die Routine benötigt zur Berechnung einer 18reihigen Determinante ca. zwei Minuten.

Die Routine

Innerhalb der drei Schleifen mit den Zählern K, I, J (Zeilen 1005–1060) werden die Faktoren S(K) für die einzelnen Unterdeterminanten berechnet. Die Zeilen 1075–1100 vertauschen zwei Reihen der Determinante, wenn der Koeffizient A(1,1) der Unterdeterminant 0 ist. A(1,1) wird in Zeile 1035 als Divisor benutzt. In Zeile 1065 werden die Faktoren S(K) multipliziert und ergeben den Wert D der Determinante.

22. Wendepunkte von rationalen Funktionen

```
100 REM INPUT
120 PRINT CHR$(147)
140 F=0: G=0: NF=0
200 OPEN1,0: IF F< >0
    THEN 300
210 PRINT" GRAD ";
220 INPUT#1,G: IF G<3
    GOTO 220
230 FOR I=G TO 0 STEP-1:
    PRINT
240 PRINT" ZAHL VOR X↑"I": ";
250 INPUT#1,K(I): NEXT:
    PRINT
300 PRINT" XA-WERT: ";
310 INPUT#1,XA
320 PRINT" XE-WERT: ";
330 INPUT#1,XE: PRINT:
    CLOSE1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT: IF NF< >0
    THEN 760
720 PRINT" WENDEPUNKT "
730 PRINT
740 PRINT" ("X"/"Y")"
750 PRINT: G$=""
760 PRINT
    "NEUE FUNKTION? J!"
770 GET G$: IF G$=""
    THEN 770
780 IF G$="J" THEN 100
790 F=1: NF=0: GOTO 200
1000 REM ROUTINE:
     WENDEPUNKT
1005 FOR L=1 TO 10: N=.1↑L
1010 IF L>1 GOTO 1025
1015 FOR X=XA TO XE+N
     STEP N
1020 GOTO 1030
1025 FOR X=Z TO Z+.2
     STEP N
1030 Z=X-N: Y=0: W=0
1035 FOR J=G TO 0 STEP -1
1040 Y=Y+J*(J-1)*K(J)*
     X↑(J-2)
1045 W=W+J*(J-1)*K(J)*
     Z↑(J-2)
1050 NEXT
1055 A=SGN(Y)+SGN(W)
1060 IF ABS(Y)<1.E-8
     THEN 1090
1065 IF W=Y THEN 1090
1070 IF A=0 THEN 1085
1075 IF X>XE THEN NF=1:
     RETURN
1080 NEXT X
1085 NEXT L
1090 FOR J=G TO 0 STEP-1
1095 Y=Y+K(J)*X↑J: NEXT
1100 RETURN
```

Der Aufruf

Die nachfolgende Routine berechnet die x- und y-Koordinate eines Wendepunktes für rationale Funktionen. Dem Unterprogramm werden ab Zeile 1000 der Grad G (Zeile 220), die Koeffizienten K(I) (Zeile 250) der Funktion und der Suchbereich mit dem Anfangswert XA und dem Endwert XE übergeben (Zeilen 310, 330).

Die Routine

Innerhalb zweier Schleifen (Zeilen 1005–1085, 1015–1080) wird der x-Wert des Wendepunktes ermittelt. In den Zeilen 1090–1100 wird die entsprechende y-Koordinate berechnet. Die Ausgabe der Wendepunkt-Koordinaten erfolgt in Zeile 740 mit den Variablen X,Y. Wird kein Wendepunkt im angegebenen Suchbereich gefunden, so wird die Flagge NF = 1 gesetzt und die Wendepunktausgabe-Routine (Zeilen 720–750) übersprungen.

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore Home-Computer, für andere können Anpassungen nötig sein.

men befinden, so wird gefragt, ob es überschrieben werden soll. Die Grafik kann nun bei Bedarf noch direkt als Paint-Magic-Bild abgespeichert oder in den Diashow/Hi-Eddi-Bereich ($2000) verschoben werden.

Wenn Paint-Magic-Bilder in Print-Shop-Bilder umgewandelt werden sollen (oder umgekehrt), so brauchen diese Grafiken nicht verschoben werden, denn beide Grafiken liegen im Computer-Speicher an der gleichen Stelle ($4000). Das gleiche gilt für Diashow/Hi-Eddi. Wenn Grafiken (zum Beispiel von Diashow) als Paint-Magic- oder Koala-Pad-Bilder abgespeichert werden sollen, so ist folgendes zu beachten: Zuerst muß mit dem Original-Programm (Paint-Magic oder Koala-Pad) ein leeres Grafikbild abgespeichert werden. Nach dem Programmstart von Bild-Copy wird nun zuerst diese leere Grafik geladen; und zwar Koala-Pad mit (1) und Paint-Magic mit (2). Anschließend wird wie gewohnt verfahren: Grafikbild laden mit (1), mit (3) verschieben und mit (4) abspeichern.

Einige Tips

1. Wenn Koala-Pad-Bilder in andere Bildformate umgewandelt werden, so treten manchmal (je nach Anzahl und Art der verwendeten Farben) Farbfehler auf.

Tip 1:
Das Koala-Pad-Bild erst in ein Paint-Magic-Bild wandeln, mit Paint-Magic die Farbfehler korrigieren und dann das Paint-Magic-Bild weiter verwandeln (zum Beispiel in Print-Shop).

2. Soll das Paint-Magic-Bild die gleichen Farben besitzen wie der Ausdruck von Print-Shop, so müssen bei Paint-Magic folgende Farben eingestellt werden:
Farbe Nr. 1 = schwarz
Farbe Nr. 2 = dunkel-grau
Farbe Nr. 3 = dunkel-grau
Farbe Nr. 4 = weiß

3. Grafiken aus Spielen
Kann das Spiel unterbrochen werden (zum Beispiel mit einem Reset-Knopf), so bleibt im allgemeinen eine Grafik im Computerspeicher erhalten. Nun Hi-Eddi laden, starten und Betriebsart

Scan: S. Höltgen

Scan: S. Höltgen

Lifegame

J. H. Conway erfand ein Spiel, das die Kultur von Punkten auf einem Rechteck simuliert. Punkte sterben, überleben oder werden geboren. Ausschlaggebend dafür ist, wieviele Nachbarpunkte der jeweilige Punkt hat. Dieses Programm für den Schneider CPC 464 und NLQ 401 rechnet ein Rechteck nach den Regeln Conways durch

Nach dem Starten können die Maße und das Rechteck mit Punkten besetzt werden. Danach wird es durchgerechnet:

N = Nochmal durchrechnen

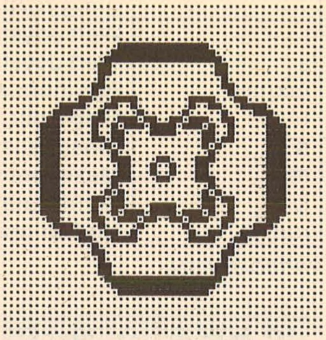
V = Verbessern, das heißt, es kann in dem Bild nochmals gemalt werden.

D = Druck eines Bildes (Ausgabe auf NLQ 401 angepaßt)

Andere Taste = Ende des Rechnens. Nach nochmaliger Abfrage kann entweder:

1. Ein Diagramm, das die Anzahl der Punkte von jedem Rechenschritt übersichtlich darstellt ausgegeben,
2. ein neues Rechteck zum Rechnen definiert oder
3. das Programm beendet werden.

Durch Durchrechnen von einfachen geometrischen Figuren (zum Beispiel Diagonalen) ergeben sich tolle



Bildschirmausdruck

Muster, die auf dem Drucker dann unverzerrt (auf dem Bildschirm ist es seitlich verzerrt) voll zur Geltung kommen.

Doch nun zum Programm: Die Hauptarbeit leistet ein MC-Programm, das ab &A000 liegt und das Bild durchrechnet (ganzer Bildschirm in 31 s). Das Unterprogramm ab 20000 legt es im Speicher ab. Hier die Auflistung der Unterprogramme: *Dieter Braun*

10000	Copyright
11000	Ausgabe von a$ in vers Farben mit Ton
12000	Erste Anleitung
13000	Eingabe von Breite und Höhe des Rechtecks. Füttert das MC-Programm mit den benötigten Daten.
14000	Bildaufbau
15000	Zweite Anleitung
16000	Punkte setzen und Striche ziehen
18000	Ausgabe auf Drucker, entweder fein (132 Zeichen/Zeile) oder grob (80 Zeichen/Zeile)
19000	Dimensionierung
20000	Ablage des MC-Programmes
21000	Diagramm ausgeben Diagrammdaten auf Drucker ausgeben.
Das Programm von 10 bis 650 verwaltet die Unterprogramme ab 10000.	

Unterprogramme

Variable	Bedeutung
p(1000)	Anzahl der Punkte
ende	Zeiger von p(1000)
b	Breite Rechteck
h	Höhe Rechteck
xx/yy	x/y Koordinate linke obere Ecke des Rechtecks

Programmbibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 12)

23. Lösen von linearen Gleichungssystemen

Die nachfolgende Routine löst lineare Gleichungssysteme beliebiger Ordnung. Einschränkungen ergeben sich aus der Länge der Eingabezeilen (beim C64 zwei Zeilen oder 80 Zeichen).

Der Aufruf

Dem Unterprogramm werden die Anzahl der Gleichungen N (Reihen) und die Koeffizienten A (I, J) des Gleichungssystems Zeile für Zeile und Spalte um Spalte übergeben. Die Dimensionierung der Felder erfolgt in Zeile 330.

Die Routine

Das Unterprogramm von Zeile 1000–1100 berechnet die Dreiecksform der Determinante nach dem Gaußschen Verfahren (siehe auch HC 5/86). Das Unterprogramm von Zeile 3000–3170 ermittelt, wenn die Hauptdeterminante ungleich Null ist, die Lösungen L(I) und gibt diese in den Zeilen 3140–3160 aus.

```
100 REM PROGRAMM GLEICHUNGEN
300 PRINT CHR$(147): CLR
310 OPEN 1,0: PRINT
320 PRINT TAB(2) "ANZAHL DER REIHEN:";
330 INPUT#1,N: DIM A(N+1,N+1), L(N): PRINT
340 PRINT: PRINT TAB(2);
350 FOR I=1 TO N
360 FOR J=1 TO N+1
370 INPUT#1,A(I,J)
380 NEXT J
390 PRINT: PRINT TAB(2): NEXTI
400 CLOSE 1
500 GOSUB 1000
700 REM LOESUNGEN AUSGABE
710 PRINT: PRINT:GOSUB 3000: PRINT
720 PRINT TAB(3) "WERT DER NENNER-DETERMI-
    NANTE":
730 PRINT: PRINT TAB(5) D: PRINT
740 IF D THEN 900
750 IF ABS(A(N,N+1))<1.E-8 THEN 800
760 PRINT: PRINT TAB(5) "KEINE LOESUNGEN!"
```

Scan: S. Höltgen


```
770 PRINT: GOTO 900
800 PRINT: PRINT TAB(5) "UNENDLICH VIELE LOE-
    SUNGEN!"
810 PRINT
900 PRINT TAB(2) "NEUE WERTE?"
910 PRINT TAB(2) "TASTE J ODER N!"
920 G$=""
930 GET G$: IF G$="J" GOTO 300
940 IF G$="N" THEN END
950 GOTO 920
1000 REM WERT DER DETERMINANTE
1005 D=1: FOR K=1 TO N-1
1015 IF A(K,K)=0 THEN GOSUB 1075
1020 IF D=0 THEN RETURN
1025 FOR J=K+1 TO N : Q=A(J,K)/A(K,K)
1030 A(J,K)=0: FOR I=K+1 TO N+1
1035 A(J,I)=A(J,I)-A(K,I) * Q
1040 NEXTI: NEXTJ
1045 NEXT K
1050 FOR I=1 TO N
1055 IF ABS(A(I,I))<1.E-8 THEN A(I,I)=0
1065 D=D * A(I,I): NEXT
1070 RETURN
1075 FOR L=K TO N
1080 IF A(L,K)<>0 THEN 1090
1085 NEXT: D=0: RETURN
1090 FOR P=1 TO N+1
1095 H=A(K,P): A(K,P)=-A(L,P)
1100 A(L,P)=H : NEXT: RETURN
3000 REM LOESUNGEN BERECHNEN
3010 FOR I=1 TO N: PRINT TAB(2);
3020 FOR J=1 TO N+1
3030 PRINT A(I,J);: NEXT
3040 PRINT: NEXT
3050 FOR I=1 TO N: L(I)=0: NEXT
3060 IF D=0 THEN RETURN
3070 FOR I=N TO 1 STEP -1
3080 S=A(I,N+1)
3090 FOR J=N TO 1 STEP -1
3100 S=S-A(I,J) * L(J)
3110 NEXT J
3120 L(I)=S/A(I,I)
3130 NEXT I: PRINT:
3140 PRINT TAB(5) " LOESUNGEN: " : PRINT
3150 FOR I=1 TO N
3160 PRINT TAB(2) L(I);: NEXT: PRINT
3170 RETURN
```

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

Variable	Bedeutung
adr	Bildschirmadresse von xx/yy
high/low	High-Byte und Low-Byte von adr
f(4)	Farben der Schrift (siehe Unterprogramm ab 11000)
x3/y3	Zwischenspeicher xx/yy
mox	Rest aus xx/4
moy	Rest aus yy/2, beides nötig, um nachher bei der beschleunigenden Cursorbewegung richtig zu runden
j	Wert Joystick (0)
jj	Wert Joystick (1)
mx/my	Additionswert bei Cursorbewegungen
t	Farbe der Cursorpos.
tt	Zwischenspeicher t (alter t-Wert)
ma	Maximum bei Diagramm
s	Schrittrate (x-Koordinaten) bei Diagramm
run	Laufvariable (for-next)

Variablenliste

```
10 GOSUB 10000:REM Copyright
90 MEMORY &9FFF
100 GOSUB 20000:KEY 139,"goto 600"+CHR$(
13)
415 FOR n=0 TO 2500:NEXT' Verzoegern
420 GOSUB 12000:REM Anleitung
425 GOSUB 19000:REM Dimensionierungen
430 GOSUB 13000:REM Init (x/y) Bildanf (
hl)
435 GOSUB 15000:REM 2. Anleitung
440 GOSUB 14000:REM BIldaufbau
445 x=0:y=0:ende=0
450 GOSUB 16000:REM Punkte malen
455 PLOT x,y,tt:GOSUB 460:GOTO 490
460 LOCATE 1,25:PRINT CHR$(24);"N";CHR$(
24);"ochmal ";CHR$(24);"D";CHR$(24);"ruc
k ";CHR$(24);"V";CHR$(24);"erb. ";
470 RETURN
480 CALL &A1E0:LOCATE 1,1:PRINT CHR$(7);
:p(ende)=PEEK(&A1FF)*256+PEEK(&A1FE)
485 ende=ende+1
490 A$=UPPER$(INKEY$):IF a$="" THEN 490
500 IF A$="N" THEN 480
510 IF A$="V" THEN GOSUB 14060:GOTO 450
515 IF a$="D" THEN IF b<133 THEN GOSUB 1
8000:GOTO 490 ELSE 490
520 LOCATE 1,25:PRINT" ENDE ? (J/N)
  ";
530 A$=UPPER$(INKEY$):IF a$="" THEN 530
540 IF a$="J" THEN 600
550 IF a$<>"N" THEN 490
560 GOSUB 460:GOTO 490
600 MODE 0:CLS:LOCATE 1,7:PRINT"1 - Diag
ramm":PRINT:PRINT:PRINT"2 - Neues
Feld"
605 PRINT:PRINT:PRINT:PRINT"3 - ENDE"
607 PRINT:PRINT:PRINT:PRINT:PRINT"Ihre W
ahl (1-3) ?"
610 a$=UPPER$(INKEY$):IF a$="" THEN 610
620 IF a$="1" THEN 21000
630 IF a$="2" THEN INK 0,1:INK 1,24:PEN
1:PAPER 0:BORDER 1:GOTO 430
635 IF a$<>"3" THEN 610
640 MODE 2:PAPER 0:PEN 1:CLS
650 END
10000 MODE 0
10010 INK 0,9:INK 1,24:PAPER 0:PEN 1
10020 INK 2,26:INK 3,6:INK 4,18:CLS
10030 a$="L I F E  G A M E":LOCATE 3,8:f
(0)=1:f(1)=2:f(2)=3:f(3)=4:ende=3
10035 GOSUB 11000
10040 LOCATE 5,12:PEN 3:PRINT"Copyright
by"
10050 PEN 4:LOCATE 5,15:PRINT"Dieter Bra
un"
10060 RETURN
11000 l=LEN(a$):x=0
11010 FOR n=1 TO l
11020 PEN f(x):PRINT MID$(a$,n,1);
11030 x=x+1:IF x=ende+1 THEN x=0
11032 SOUND 1,700-n*24,20,7
11035 WHILE (SQ(1) AND &80)=&80:WEND
11040 NEXT
11050 RETURN
12000 REM Anleitung
12010 MODE 1:INK 0,1:INK 1,24:BORDER 1:P
EN 1:PAPER 0:CLS
```

Scan: S. Höltgen

```
43 REM* BILDSCHIRM-UEBERSCHRIFT
44 PRINTCHR$(14):PRINT"■";
45 PRINT"■";
46 PRINT"            -EMONSTRATIONSPROGRAMM
   ";
47 FOR X=0 TO 39
48 PRINT"=";
49 NEXT X
50 PRINT"■"
51 :
52 REM* INFORMATION
53 PRINT"■■\IT DER NEUEN '-NPUT'-_OUTINE
 STEHEN"
54 PRINT"■■■-HNEN VIELE NEUE  \OEGLICHKE
ITEN DER"
55 PRINT"■■■PROFFESIONELLEN -ROGRAMMIERU
NG EIGE-"
56 PRINT"■■■NER XERWALTUNGS--ROGRAMME OF
FEN.  "
57 PRINT"■■■■■-IE XORTEILE LIEGEN AUF DE
R |AND: "
58 PRINT"■■■■■- |ILDSCHIRMMASKEN UNZERST
OERBAR"
59 PRINT"■■■■■- -ATENAUSGABE WIRKT DURCH
 GLEICHE"
60 PRINT"■■■■■■■AENGEN DER ♥TRINGS GEORD
NETER   "
61 PRINT"■■■■■- STOERENDE _RAGEZEICHEN K
OENNEN  "
62 PRINT"■■■■■■UNTERDRUECKT WERDEN
   "
63 PRINT"■■■■■■■■■■■■■■■ETC."
64 PRINT"■■-RUECKEN ♥IE _-| /_/ FUER EIN
|EISPIEL"
65 GET X$:IF X$<>CHR$(13)THEN 65
66 :
67 REM* BEISPIEL
68 PRINT"■"
69 PRINT"■ |EBEN ♥IE BITTE IHREN VOLLEN
/AMEN EIN"
70 PRINT"■■■■■XORNAME :";
71 LN=15:REM* MAX. LAENGE
72 GOSUB 8
73 VN$=A$
74 PRINT"■■■■■/ACHNAME:";
75 LN=20:REM* MAX. LAENGE
76 GOSUB 8
77 NN$=A$
78 :
79 REM* TESTWEISE AUSGABE
80 PRINT"■■♣USGABE :■■"
81 PRINT" "VN$,NN$
82 :
83 REM* ENDE
84 PRINT"■■■■XIEL ♥PASS MIT DER NEUEN _O
UTINE !!!"
85 PRINT"■■■■        _-| /_/ FUER -/--"
86 GET X$:IF X$<>CHR$(13) THEN 86
87 PRINT"■"CHR$(142)
88 END
READY.
```


Programm-bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 13)

24. Mittelwert und mittlerer Fehler

Die nachfolgende Routine berechnet für eine Anzahl von Zahlenwerten den Mittelwert, den mittleren Fehler des Mittelwertes und eines Einzelwertes.

Der Aufruf

Dem Unterprogramm werden ab Zeile 1000 eine Anzahl N von Zahlen X(I) übergeben. Vor der Übergabe erfolgt die Anlegung des Feldes durch eine DIM-Anweisung in Zeile 230. Das Programm berechnet jetzt die Variablen XM (Mittelwert), MX (Mittlerer Fehler des Mittelwertes) und M (Mittlerer Fehler eines Einzelwertes). Diese Werte werden in den Zeilen 720, 750 und 780 ausgegeben.

Die Routine

In Zeile 1030 wird der Mittelwert aus der Summe SX und der Anzahl der Werte gebildet. Die Schleife von Zeile 1035 bis 1050 berechnet die Abweichung vom Mittelwert, wobei die Liste X(I) der ursprünglichen Werte verlorengeht, weil die berechneten Abweichungen wieder dem Feld X(I) zugewiesen werden. Zeile 1045 berechnet die Summe der Fehlerquadrate. Zeile 1055 ermittelt den mittleren Fehler eines Einzelwertes, Zeile 1060 den mittleren Fehler des Mittelwertes.

```
100 REM INPUT
120 PRINT CHR$(147)
200 CLR: OPEN 1,0
210 REM EINGABEN
220 PRINT "ANZAHL DER WERTE   ";
230 INPUT#1,N : DIM X(N)
240 FOR I=1 TO N
250 PRINT
260 PRINT I".WERT:   ";
270 INPUT#1,X(I)
280 NEXT
290 CLOSE 1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT: PRINT
720 PRINT TAB(2) "MITTELWERT   ";XM
730 PRINT: PRINT
740 PRINT TAB (2) "MITTLERER FEHLER DES"
750 PRINT TAB(2) "MITTELWERTES   "; MX
760 PRINT: PRINT
770 PRINT TAB(2) "MITTLERER FEHLER
780 PRINT TAB(2) "EINES WERTES   "; M
790 PRINT: PRINT: G$=""
800 GET G$: IF G$="" GOTO 800
810 GOTO 200: REM END
1000 REM ROUTINE
1005 SX=0: S1=0
1010 IF N<2 THEN RETURN
1015 FOR I=1 TO N
1020 SX=SX+X(I)
1025 NEXT I
1030 XM=SX/N
1035 FOR I=1 TO N
1040 X(I)=X(I)-XM
```

Scan: S. Höltgen

```
1045 S1=S1+X(I)*X(I)
1050 NEXT I
1055 M=SQR(S1/(N-1))
1060 MX=M/SQR(N)
1065 RETURN
```

25. Ausgleichsgeraden

Die nachfolgende Routine berechnet aus der Anzahl von zusammengehörigen Wertepaaren die optimale lineare Funktion (Ausgleichsgerade).
Bei zwei Wertepaaren ergibt sich die Zweipunktform der Geradengleichung.

Der Aufruf

Dem Unterprogramm (ab Zeile 1000) werden die Anzahl N und die Paare X(I), Y(I) übergeben. In Zeile 230 erfolgt eine entsprechende Felddimensionierung. Das Programm ermittelt für die lineare Funktion y = a + b * x die Konstanten a, b nach der Methode der kleinsten Fehlerquadrate. Die Gleichung der linearen Funktion wird in der Variablen L$ gespeichert und in Zeile 760 ausgegeben.

Die Routine

Die erste Schleife (Zeile 1010 bis 1025) berechnet die Summen der Variablen und den Mittelwert MX. Die zweite Schleife berechnet die Summen der Fehlerquadrate, aus denen in Zeile 1060 die Konstanten A,B berechnet werden. Zeile 1065 bis Zeile 1080 bilden die lineare Funktion im String L$.

```
100 REM INPUT
120 CLR: PRINT CHR$(147)
200 OPEN 1,0
210 REM EINGABEN
220 PRINT " ANZAHL DER PAARE ";
230 INPUT#1,N: DIM X(N), Y(N)
240 FOR I=1 TO N
250 PRINT
260 PRINT I".PAAR:   ";
270 INPUT#1,X(I),Y(I)
280 NEXT
290 CLOSE 1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT: PRINT: PRINT
720 IF S2 GOTO 760
730 PRINT "KEINE LOESUNG!"
740 PRINT: PRINT: PRINT
750 GOTO 780
760 PRINT TAB(2) L$
770 PRINT: PRINT: PRINT
780 PRINT TAB(7) "TASTE!"
790 PRINT: PRINT: G$=""
800 GET G$: IF G$="" GOTO 800
810 GOTO 120: REM END
1000 REM
1005 SX=0: SY=0: S1=0: S2=0
1010 FOR I=1 TO N
1015 SY=SY+Y(I)
1020 SX=SX+X(I)
1025 NEXT I: MX=SX/N
1030 FOR I=1 TO N
1035 X(I)=X(I)-MX
1040 S1=S1+X(I)*Y(I)
1045 S2=S2+X(I)*X(I)
1050 NEXT I
1055 IF S2=0 THEN RETURN
1060 B=S1/S2: A=SY/N-B*MX
1065 V$=" +"
1070 IF SGN(B)<0 THEN V$="-"
1075 L$="Y ="+STR$(A)+V$
1080 L$=L$+STR$(ABS(B))+" * X"
1085 RETURN
```

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

OLD mit einem einzigen POKE

Mit einem einzigen POKE können bereits verloren geglaubte Programme zurückgeholt werden (C64).

Hat man ein Programm versehentlich mit NEW gelöscht, dann tippt man folgenden POKE ein:
POKE 2050,X
Wobei X mindestens um eins höher ist, als es die erste Zeilennummer war.
Beispiel:
10 Print „Dies ist ein Test"
20 Print „Der Ein-Poke-Routine"
30 Rem
NEW (Return)
POKE 2050,11 (Return)
11 (Return)
LIST (Return)
Nun erscheint das Listing.
Achtung: Tippt man nun noch eine Zeile ein, hängt sich der Rechner auf. Bei relativ kurzen Programmen, die noch ganz auf den Bildschirm passen, tippt man NEW (Return) und fährt mit dem Cursor auf die Zeilen, die sich noch auf dem Bildschirm befinden und drückt immer wieder die Returntaste. Nach LIST befindet sich das Programm wieder im Speicher.

Klaus Hegemann

Borderlines

Dieses Programm für den C64 plus Simons BASIC zeigt, wie man mit einfachen mathematischen Funktionen phantastische bunte Bilder auf dem Monitor erstellen kann.

Das Motiv wird durch vier Plot-Anweisungen aus der Bildmitte heraus entwickelt. Es können jeweils eine Funktion in X-Richtung und eine in Y-Richtung gezeichnet werden. Dabei kann man beide Bilder getrennt entstehen lassen oder überlagert das X-Bild mit dem Y-Bild.
Die waagerechte Funktion hat zunächst eine Amplitude von 30 Pixeln. A wird pro Zeile um −1 vermindert, durchläuft den Nullpunkt – die Funktion wird zu einer Geraden – und endet bei A = −70. Gleichzeitig wird die Periode F mit dem Anfangswert 1 jeweils um 0,01 vermindert, so daß sie am oberen und unteren Bildrand den Wert 0 hat. Einen zusätzlichen Pfiff erhalten die Bilder durch laufende Veränderung der Phase P.
Die Veränderungen gelten auch für das Motiv in Y-Richtung, nur daß aufgrund der unterschiedlichen Koordinaten die Endwerte anders liegen. R = MOD(YA,4) steuert die vier Farben in Abhängigkeit der Funktion.
Mit RUN erscheint das Menü auf dem Bildschirm. Nach Eingabe von 1 ⟨RETURN⟩ sieht man die beiden Funktionszeilen 100 und 110, und die gewünschten Funktionen können eingefügt werden. Danach startet das Programm nach Drücken der Funktionstaste f1 mit dem Zeichen der Funktion in X-Richtung (zirka 15 min). Ist das Bild fertig, so erscheint das Menü wieder auf dem Monitor. Jetzt wird entweder 2 oder 3 eingegeben, je nachdem ob die zweite Funktion getrennt oder überlagert aufgebaut werden soll.
Mit 5 wird das Bild zurückgeholt, mit 4 das Motiv in allen Farben gezeigt. Es entstehen dabei herrliche Farbeffekte.
Zur Erzeugung der Bilder können alle möglichen Winkelfunktionen verwendet werden. Zum Beispiel 3*sin(1−X), TAN(X)12, 1/TAN(X12), SIN(X)+ATN(X) und so weiter. Im Handbuch sind viele Funktionen aufgeführt. Ferner sollte man auch Potenzfunktionen ausprobieren: SQR(X-PI/2), X1−2, X1(1/3) und so weiter. Auch kann man LOG(X), EXP(X) oder Ausdrücke wie X↑COS(X) verwenden. Läßt man das Bild mit dem Nullpunkt an der linken oberen Ecke mit PLOTX,Y,R (Koor-

Scan: S. Höltgen

```
BE 03 20 09 E5 21 B8 7F --327
CB CE E1 18 2A 23 23 05 --307
05 7E FE 23 28 0E FE 21 --2F9
28 0D FE 40 28 0D 23 10 --1DB
F0 C3 B9 C4 AF 18 06 3E --43B
01 18 02 3E 02 32 EF 7F --1FB
3E 20 2B BE C2 D1 C4 3A --3D8
FC 7F DD BE 04 28 0D DD --42C
BE 05 28 08 DD BE 06 28 --2BC
03 C3 B9 C4 3E 01 DD BE --41D
03 CA 1D C4 3A EF 7F DD --433
BE 07 CA 1D C4 DD BE 08 --413
CA 1D C4 DD BE 09 CA 1D --436
C4 3E 02 FD E1 FB CD F7 --5A1
C6 3E 05 FD E1 FB CD F7 --5A6
C6 3E 06 FD E1 FB CD F7 --5A7
C6 FD E1 3E 07 FB CD F7 --5A8
C6 F3 FD E5 FD 21 D7 7F --60F
DD 2A F6 7F DD 29 DD 29 --488
11 00 40 DD E5 E1 19 CD --3DA
C0 DA A7 ED 52 E5 DD E1 --623
11 40 9C DD 19 3A FD 7F --399
A7 CA 31 C6 06 04 CB 27 --364
10 FC 5F 3A FC 7F CB 27 --412
CB 27 B3 5F 3A EF 7F B3 --45F
5F DD 73 00 3A FC 7F CD --431
56 C7 06 11 21 DA 7F BE --36C
CA 2E C5 23 10 F9 23 3E --34A
2D BE C2 3E C5 36 30 E5 --3FB
21 B8 7F CB C6 E1 06 05 --3D5
11 C9 7F 7E FE 20 12 23 --32A
13 CA 4E C5 10 F5 CD 30 --3F2
C7 CD 69 C7 CD 75 C7 3A --507
FC 7F A7 CA 6E C5 21 FF --53F
05 ED 5B C5 7F A7 ED 52 --477
DA C9 C4 C3 7B C5 21 FF --58A
0F ED 5B C5 7F A7 ED 52 --481
DA C9 C4 21 B8 7F CB 46 --4D0
CA 99 C5 CB 86 3A FC 7F --52E
A7 CA C9 C4 ED 5B C5 7F --58A
21 00 06 A7 ED 52 22 C5 --2F4
7F 2A C5 7F DD 75 01 CB --40B
24 CB 24 CB 24 CB 24 DD --3CE
74 02 21 B8 7F CB 4E CB --3B2
8E C2 1D C4 3A FC 7F CD --4B3
56 C7 06 11 21 DA 7F BE --36C
CA C6 C5 23 10 F9 23 3A --3DE
EF 7F CD 56 C7 BE CA D4 --5B4
C5 23 10 F9 23 3E 2D BE --33D
C2 E4 C5 E5 21 B8 7F CB --573
C6 E1 36 30 06 05 11 C9 --2F2
7F 7E FE 20 12 23 13 CA --32D
F4 C5 10 F5 CD 30 C7 CD --54F
69 C7 CD 75 C7 21 FF 05 --45E
ED 5B C5 7F A7 ED 52 DA --54C
C9 C4 21 B8 7F CB 46 CA --4C0
21 C6 CB 86 ED 5B C5 7F --4C4
21 00 06 A7 ED 52 22 C5 --2F4
7F 2A C5 7F DD 75 03 DD --41F
7E 02 B4 DD 77 02 C3 1D --36A
C4 3E 64 FD BE 00 28 12 --35B
2A F6 7F 2B 3E FF BC C2 --485
44 C6 26 05 22 F6 7F C3 --38F
1D C4 06 11 21 DA 7F 3E --2B0
20 BE C2 58 C6 23 10 F9 --3EA
3E 2D BE C2 67 C6 E5 21 --41E
B8 7F CB C6 E1 36 30 06 --415
0B 11 C9 7F 7E FE 20 12 --312
CA 77 C6 23 13 10 F5 CD --40F
30 C7 CD 69 C7 CD 75 C7 --4FD
21 B8 7F CB 46 CA B9 C6 --4B2
CB 86 21 00 80 ED 5B C7 --401
7F A7 ED 52 DA C9 C4 A7 --573
ED 5B C5 7F 21 00 00 ED --39A
```


Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 14)

26. Quadratische Regression (Ausgleichsparabel)

Die nachfolgende Routine berechnet aus einer Vielzahl N von zusammengehörigen Wertepaaren die optimale quadratische Funktion (Ausgleichsparabel). Bei drei Wertepaaren, die nicht auf einer Geraden liegen, ergibt sich eine quadratische Gleichung.

```
100 REM INPUT
120 CLR : PRINT CHR$(147)
200 OPEN 1,0
210 REM EINGABEN
220 PRINT " ANZAHL DER PAARE ";
230 INPUT#1,N: DIM X(N), Y(N)
240 FOR I=1 TO N
250 PRINT
260 PRINT I".PAAR:   ";
270 INPUT#1,X(I),Y(I)
280 NEXT
290 CLOSE 1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT: PRINT: PRINT
720 IF D=0 GOTO 760
730 PRINT TAB(2) L$ : PRINT
740 PRINT TAB(2) "A=" A : PRINT TAB(2) "B=" B
750 PRINT TAB(2) "C=" C : GOTO 770
760 PRINT " KEINE LOESUNG!"
770 PRINT: PRINT: PRINT
780 PRINT TAB(7) "TASTE!"
790 PRINT: PRINT: G$=""
800 GET G$: IF G$="" GOTO 800
810 GOTO 120: REM END
1000 REM
1005 FOR I=1 TO 4 : S(I)=0 : R(I)=0
1010 NEXT I : FOR J=1 TO 4
1015 FOR I=1 TO N : S(J)=S(J)+X(I)↑J
1020 NEXT I : NEXT J
1025 FOR J=1 TO 3
1030 FOR I=1 TO N
1035 R(J)=R(J)+(X(I)↑(J-1))*Y(I)
1040 NEXT I : NEXT J
1045 D=N*(S(2)*S(4)-S(3)↑2)+2*S(1)*S(2)*S(3)
1050 D=D-S(2)↑3-S(4)*S(1)↑2
1055 IF ABS(D)<2.E-8 THEN D= 0 : RETURN
1060 A=R(1)*(S(2)*S(4)-S(3)↑2)+R(3)*(S(1)*S(3)-S(2)↑2)
1065 A=A+R(2)*(S(2)*S(3)-S(1)*S(4))
1070 B=R(2)*(N*S(4)-S(2)↑2)+R(1)*(S(2)*S(3)-S(1)*S(4))
1075 B=B+R(3)*(S(1)*S(2)-N*S(3))
1080 C=R(3)*(N*S(2)-S(1)↑2)+R(2)*(S(1)*S(2)-N*S(3))
1085 C=C+R(1)*(S(1)*S(3)-S(2)↑2)
1090 C=C/D : B=B/D : A=A/D
1095 L$="Y = A + B * X + C * X * X"
1100 RETURN
```

Der Aufruf

Dem Unterprogramm werden ab Zeile 1000 die Anzahl N und die Paare (X(I), Y(I)) übergeben. Die DIM-Anweisung erfolgt in Zeile 230. Das Programm ermittelt für die quadratische Funktion

$$y = a + bx + cx^2$$

die Konstanten a, b, c nach der Methode der kleinsten Fehlerquadrate. Die Werte dieser Konstanten werden in den Variablen A, B und C gespeichert. Diese werden in Zeile 740 bzw. 750 ausgegeben.

Scan: S. Höltgen

Die Routine

Zeile 1015 bildet mit den dazugehörigen Schleifen die Summe der Fehlerquadrate. Aus diesen Summen werden in den Zeilen 1060 bis 1085 Zwischenwerte berechnet. In Zeile 1090 erfolgt die Bestimmung der Konstanten A, B und C.

27. Korrelationskoeffizient

Die Korrelationsrechnung wird in der Statistik benutzt, um bei Kollektiven zwischen zwei charakteristischen Eigenschaften Zusammenhänge feststellen zu können. Die gegenseitige, mehr oder weniger feste Abhängigkeit dieser zwei Merkmale des Kollektivs wird zahlenmäßig durch den Korrelationskoeffizienten beschrieben. Die Werte liegen zwischen −1 (kein Zusammenhang) und +1 (starker Zusammenhang). Je näher der Korrelationskoeffizient am Wert 1 liegt, desto sicherer ist der vermutete Zusammenhang.

```
100 REM INPUT
120 CLR: PRINT CHR$(147)
200 OPEN 1,0
210 REM EINGABEN
220 PRINT " ANZAHL DER PAARE ";
230 INPUT#1,N: DIM X(N), Y(N)
240 FOR I=1 TO N
250 PRINT
260 PRINT I".PAAR:  ";
270 INPUT#1,X(I),Y(I)
280 NEXT
290 CLOSE 1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT: PRINT: PRINT
720 IF S2>0 AND S3>0 THEN GOTO 760
730 PRINT"  KEINE LOESUNG!"
740 PRINT: PRINT: PRINT
750 GOTO 780
760 PRINT TAB(2) L$
770 PRINT: PRINT TAB(4) "R=" R
780 PRINT: PRINT TAB(7) "TASTE!"
790 PRINT: PRINT: G$=""
800 GET G$: IF G$="" GOTO 800
810 GOTO 120: REM END
1000 REM
1005 SX=0: SY=0: S1=0: S2=0: S3=0
1010 FOR I =1 TO N
1015 SY=SY+Y(I)
1020 SX=SX+X(I)
1025 NEXT I: MX=SX/N : MY=SY/N
1030 FOR I=1 TO N
1035 X(I)=X(I)-MX : Y(I)=Y(I)-MY
1040 S1=S1+X(I)*Y(I)
1045 S2=S2+X(I)*X(I)
1050 S3=S3+Y(I)*Y(I) : NEXT I
1055 IF S2=0 OR S3=0 THEN RETURN
1060 R=S1/SQR(S2*S3)
1065 L$=" KORRELATIONSKOEFFIZIENT "
1070 RETURN
```

Der Aufruf

Dem Unterprogramm werden ab Zeile 1000 die Anzahl N der Paare und die Werte der einzelnen Paare (X(I), Y(I)) übergeben. Die entsprechende DIM-Anweisung erfolgt in Zeile 230. Die Routine berechnet den Korrelationskoeffizienten in der Variablen R.

Die Routine

Die Zeilen 1005 bis 1025 dienen der Mittelwertbildung der X(I)- bzw. Y(I)-Werte. In Zeile 1035 erfolgt die Berechnung der Summe der Abweichungen der einzelnen Werte vom Mittelwert. Aus diesen Größen wird in Zeile 1060 der Korrelationskoeffizient in der Variablen R berechnet. Der Wert von R wird in Zeile 770 ausgegeben.

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

```
52 DD 75 00 DD 74 01 ED --3E3
5B C7 7F 21 00 00 ED 52 --301
DD 75 02 DD 74 03 C3 1D --388
C4 2A C5 7F DD 75 00 DD --461
74 01 2A C7 7F DD 75 02 --339
```

```
Hexdump von interp
C3 1D CF F5 FB DB FE FE --676
FF C2 17 CF F3 F1 C9 ED --641
7B FE 7F FB C9 ED 73 FE --61A
7F 31 00 7D 06 14 76 10 --1CD
FD AF 32 B7 7F 2A F6 7F --4B3
CD C0 DA 7E CD 9A CF 32 --54D
FD 7F AF CB 39 CB 1F CB --4E4
39 CB 1F 06 06 CB 3F 10 --249
FC 32 EF 7F 79 32 FC 7F --4C2
23 7E 32 FA 7F 23 23 7E --310
32 F8 7F 2B 7E CD 9A CF --488
32 FB 7F 79 32 F9 7F 3A --409
B7 7F EE FF 32 B7 7F CD --558
AA CF 2A F6 7F 23 23 23 --381
23 22 ED 7F CD BD D1 2A --436
ED 7F ED 5B F4 7F 22 F4 --53D
7F ED 53 F6 7F CD 0B CF --4DB
3A B6 7F A7 C4 6D D2 C3 --4DC
2D CF 4F AF 06 04 CB 21 --2F0
17 10 FB 06 04 CB 39 10 --240
FC C9 3A FD 7F FE 01 CA --544
D8 CF FE 02 CA 13 D0 FE --552
03 CA 56 D0 FE 04 CA 99 --458
D0 FE 05 CA AF D0 FE 06 --520
CA E7 D0 FE 07 CA 20 D1 --541
FE 08 CA 67 D1 C3 17 CF --4B1
CD EB D1 CD 2C D2 3A FC --58A
7F FE 00 C2 F0 CF ED 4B --536
F2 7F 11 00 00 C3 FD CF --411
2A F2 7F CD C0 DA 4E 23 --473
46 23 5E 23 56 3A EF 7F --2E8
FE 00 CA 17 CF 2A F0 7F --447
CD C0 DA 71 23 70 23 73 --401
23 72 C9 CD EB D1 CD 2C --4E0
D2 3A FC 7F FE 00 C2 2B --472
D0 ED 4B F2 7F 11 00 00 --38A
C3 38 D0 2A F2 7F CD C0 --4F3
DA 4E 23 46 23 5E 23 56 --28B
3A EF 7F FE 00 CA 17 CF --456
2A F0 7F CD C0 DA 7E 81 --4FF
77 23 7E 88 77 23 7E 8B --343
77 23 7E 8A 77 C9 CD EB --49A
D1 CD 2C D2 3A FC 7F FE --54F
00 C2 6E D0 ED 4B F2 7F --4A9
11 00 00 C3 7B D0 2A F2 --33B
7F CD C0 DA 4E 23 46 23 --3C0
5E 23 56 3A EF 7F FE 00 --37D
CA 17 CF 2A F0 7F CD C0 --4D6
DA 7E 91 77 23 7E 98 77 --410
23 7E 9B 77 23 7E 9A 77 --365
C9 CD EB D1 3A FC 7F FE --605
01 C2 17 CF 2A F2 7F 2B --3CF
2B 2B 2B 22 F6 7F C9 CD --3AE
EB D1 CD 2C D2 3A FC 7F --53C
FE 01 C2 17 CF 3A EF 7F --44F
FE 00 CA 17 CF 2A F0 7F --447
CD C0 DA 4E 23 46 78 B1 --447
C2 E6 D0 23 4E 23 46 78 --3CA
B1 C2 E6 D0 2A F2 7F 2B --4EF
2B 2B 2B 22 F6 7F C9 CD --3AE
EB D1 CD 2C D2 3A FC 7F --53C
FE 01 C2 17 CF 3A EF 7F --44F
FE 00 CA 17 CF 2A F0 7F --447
CD C0 DA 4E 23 46 78 B1 --447
C2 15 D1 23 4E 23 46 78 --2FA
```

Scan: S. Höltgen

```
=0 TO gr/2: PLOT x+m+rg+h,y-c-j:
 NEXT j: NEXT h: LET a=a-b: RETU
RN
1310 DATA 76,12,76,28,76,44,76,6
0,76,76,60,76,44,76,28,76,12,76,
12,92,12,108,28,108,44,108,60,10
8,76,108
1320 DATA 76,124,76,140,76,156,7
6,172,92,172,108,172,108,156,108
,140,108,124,108,108,124,108,140
,108,156,108,172,108
1330 DATA 172,92,172,76,156,76,1
40,76,124,76,108,76,108,60,108,4
4,108,28,108,12,92,12
1340 DATA 92,28,92,44,92,60,92,7
6,28,92,44,92,60,92,76,92,92,156
,92,140,92,124,92,108,156,92,140
,92,124,92,108,92
1350 DATA 12,12,28,12,12,28,28,2
8,12,172,28,172,12,156,28,156,15
6,172,172,172,156,156,172,156,15
6,28,172,28,156,12,172,12
1400 REM Wuerfel
1410 GO SUB 1520: LET wu=INT (RN
D*6+1): PLOT 221,6: DRAW 32,0: D
RAW 0,32: DRAW -32,0: DRAW 0,-32
1415 BEEP .01,1: BEEP .01,2: BEE
P .01,1
1420 IF wu=1 OR wu=5 OR wu=3 THE
N PRINT AT 19,29;"■"
1430 IF wu<>1 THEN PRINT AT 18,2
8;"■";AT 20,30;"■"
1440 IF wu=4 OR wu=5 OR wu=6 THE
N PRINT AT 20,28;"■";AT 18,30;"■
"
1450 IF wu=6 THEN PRINT AT 18,29
;"■";AT 20,29;"■"
1460 RETURN
1500 REM Loeschen
1510 OVER 0: FOR n=0 TO 21: PRIN
T AT n,23;"         ": NEXT n: R
ETURN
1520 OVER 0: FOR n=17 TO 21: PRI
NT AT n,23;"         ": NEXT n:
RETURN
1600 REM
1610 RESTORE 1600: LET x=0: LET
y=0: LET nr=0: FOR k=1 TO 22: RE
AD ms: LET m$=CHR$ ms: LET co=15
360+CODE m$*8: GO SUB 1250: LET
y=y+8: NEXT k
1620 DATA 32,90,88,32,97,101,114
,103,101,114,116,32,109,105,99,1
04,32,127,32,77,83,32
1630 OVER 1: FOR k=0 TO 21: PRIN
T AT k,0;" ": NEXT k: OVER 0: RE
TURN
1700 REM Farbe
1710 RESTORE 1710+qw
1711 DATA 0
1712 DATA 6
1713 DATA 4
1714 DATA 3
1720 READ col
1730 RETURN
1800 REM PRINT
1810 LET qw=q: GO SUB 1700: REST
ORE 1800: GO SUB 1500
1820 DATA "ist am","Zug"
1830 PRINT AT 1,23; INK col;"████
█████": FOR k=1 TO 2: READ m$:
PRINT AT 1+k,23;m$: NEXT k
1840 RETURN
4000 REM Spieler
```


Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 15)

28. Grafische Integration

Die nachfolgende Routine berechnet die Größe eines Flächenstückes, das zwischen einer Kurve und einer waagerechten Achse liegt. Aus der grafischen Darstellung wird, bei einer beliebigen unteren Grenze beginnend, eine ungerade Zahl von Hochwerten (y-Werte) in aufeinanderfolgenden, gleichweiten Schritten abgelesen. Je kleiner die Schrittweite gewählt wird, desto genauer ist die berechnete Fläche.

Der Aufruf

Dem Unterprogramm werden ab Zeile 1000 der Anfangswert U, die Schrittweite S, die ungerade Anzahl N und die Größe der abgelesenen Hochwerte Y (I) übergeben. Vor der Eingabe der Hochwerte erfolgt durch die Anlegung der für die Berechnung notwendigen Felder durch eine DIM-Anweisung (Zeile 160). Das Programm berechnet in der Variablen F die Größe des Flächenstückes in den entsprechenden Grenzen.

Die Routine

Die Berechnung der Größe der Fläche erfolgt über die sog. M-Formel der Simpson-Regel in Zeile 1070. Die dafür notwendigen Summen und Differenzen werden im Unterprogramm berechnet.

```
110 PRINT CHR$(147)
120 CLR
130 INPUT "UNTERE GRENZE";U
140 INPUT "SCHRITTWEITE";S
150 INPUT "ANZAHL DER WERTE";N
160 DIM D1(N),D2(N),Y(N)
210 OPEN 1,0
220 FOR I=0 TO N-1
230 X=U+I*S : PRINT :
240 PRINT I+1".WERT: "X" ";
250 INPUT#1,Y(I)
260 NEXT
270 CLOSE 1
500 GOSUB 1000
720 PRINT : PRINT
740 PRINT "FLAECHE:"F
760 PRINT
780 PRINT "TASTE !" : G$=""
800 GET G$ : IF G$="" THEN 800
820 PRINT
840 GOTO 120
1005 S1=0 : S2=0 : F=0 : N=N-1
1010 FOR I=1 TO N-1 STEP 2
1015 S1=S1+Y(I)
1020 NEXT
```

Scan: S. Höltgen

```
1025 FOR I=1 TO N
1030 D1(I)=Y(I)-Y(I-1)
1035 NEXT
1040 FOR I=1 TO N-1
1045 D2(I)=D1(I+1)-D1(I)
1050 NEXT
1055 FOR I=1 TO N-1 STEP 2
1060 S2=S2+D2(I)
1065 NEXT
1070 F=2*S*(S1+S2/6)
1075 RETURN
```

Die Routine

Das Intervall (U, O) wird in n Streifen der Breite S zerlegt (Zeile 1005). In Zeile 1020 erfolgt eine Korrektur der Streifenbreite S, wenn die Grenzen nicht ganzzahlig sind. Die Berechnung der Funktionswerte erfolgt in der Schleife ab Zeile 1020 bis Zeile 1030. Aus der Streifenbreite S der Zerlegung (siehe auch Zeile 1020) und den Summen S1, S2 erfolgt die Berechnung des Wertes für das Integral in Zeile 1095 in der Variablen F.

29. Bestimmtes Integral

Die nachfolgende Routine berechnet die Größe des bestimmten Integrals von beliebigen Funktionen mit einer Variablen.

Der Aufruf

Vor dem Sprung ins Unterprogramm und vor Übergabe der Werte muß eine CLR-Anweisung erfolgen. Dem Unterprogramm werden ab Zeile 1000 die obere sowie untere Grenze O, U und der Funktionsterm F(x), (z.B. ist F(x) = SIN (x) übergeben. Der berechnete Wert des Integrals wird in Zeile 770 ausgegeben.

```
110 PRINT CHR$(147)
120 CLR
130 DEF FNF(X)=SIN(X)
140 INPUT "OBERE GRENZE";O
150 INPUT "UNTERE GRENZE";U
500 GOSUB 1000
700 REM OUTPUT
760 PRINT
770 PRINT "FLAECHE: "F
780 PRINT
790 PRINT "TASTE !" : G$=""
800 GET G$ : IF G$="" THEN 800
820 PRINT
840 GOTO 120
1005 S=0.02 : N=INT((O-U)/S+0.1)
1010 IF N/2=INT(N/2) THEN N=N+1
1015 DIM D1(N),D2(N),Y(N),X(N)
1020 S=(O-U)/(N-1) : FOR I=0 TO N
1025 X(I)=U+I*S : Y(I)=FNF(X(I))
1030 NEXT
1035 S1=0 : S2=0 - F=0 : N=N-1
1040 FOR I=1 TO N-1 STEP 2
1045 S1=S1+Y(I) : NEXT
1050 FOR I=1 TO N
1055 D1(I)=Y(I)-Y(I-1)
1060 NEXT
1065 FOR I=1 TO N-1
1070 D2(I)=D1(I+1)-D1(I)
1075 NEXT
1080 FOR I=1 TO N-1 STEP 2
1085 S2=S2+D2(I)
1090 NEXT
1095 F=2*S*(S1+S2/6)
1100 RETURN
```

```
4005 GO SUB 9100: LET re=wq
4010 GO SUB 1500: GO SUB 1800
4020 IF wq=0 THEN RETURN
4030 GO SUB 1400: LET wq=wq-1
4035 IF re=3 AND wu<>6 THEN PAUS
E 5: PAUSE 100: GO TO 4020
4100 DATA "Mit","welcher","Figur
","wollen","Sie","ziehen ?"
4110 RESTORE 4100: FOR k=1 TO 6:
 READ m$: PRINT AT 5+k,23;m$: NE
XT k
4115 LET wq=0+(wu=6): LET re=0
4120 LET m$=INKEY$: IF CODE m$<4
8 OR CODE m$>52 THEN GO TO 4120
4125 IF m$="0" THEN GO TO 6026
4130 BEEP .05,2: LET a=VAL m$: I
F s(q,a)>56 AND wu<>6 THEN GO TO
 4900
4140 IF f(q*10-9)=q AND f(q*10-9
+wu)<>q AND s(q,a)<>q*10-9 THEN
GO TO 4900
4145 RESTORE 6170: FOR k=1 TO q:
 READ c: NEXT k
4151 IF f(s(q,a)+wu)=q AND (s(q,
a))<=40 AND (s(q,a))+wu>40 AND c
=40 THEN GO TO 4900
4152 IF f(s(q,a)+wu)=q AND s(q,a
)+wu<=40 AND (((s(q,a))<c AND (s
(q,a))+wu<c) OR ((s(q,a))>c AND
(s(q,a))+wu>c)) THEN GO TO 4900
4160 LET pl=0: LET b=0: FOR k=1
TO 4: LET b=b+(f(q*4+k+52)=q): N
EXT k: IF wu=6 AND f(q*10-9)<>q
AND s(q,a)<57 AND b<>0 THEN GO T
O 4900
4170 IF s(q,a)>56 THEN LET pl=q*
10-9
4175 IF s(q,a)+wu>40 AND s(q,a)<
=40 AND q<>1 THEN LET pl=s(q,a)+
wu-40
4176 IF pl<>0 THEN GO TO 4200
4185 IF c=40 AND s(q,a)+wu<=44 T
HEN LET pl=s(q,a)+wu: GO TO 4200
4186 IF s(q,a)<=c AND s(q,a)+wu>
c THEN LET wu1=s(q,a)-c+wu: LET
pl=4*q+36+wu1: GO TO 4200
4190 IF s(q,a)>=4*q+37 AND s(q,a
)+wu<=4*q+40 THEN LET pl=s(q,a)+
wu: GO TO 4200
4195 IF s(q,a)+wu<=40 THEN LET p
l=s(q,a)+wu
4196 IF pl=0 THEN GO TO 4900
4200 IF f(pl)=q AND pl>40 OR pl>
q*4+40 THEN GO TO 4900
4201 LET nr=a: LET nr1=nr: LET a
pl=s(q,nr): IF pl>40 AND apl<=40
 THEN LET d(q)=d(q)+1
4202 IF apl>56 THEN LET w(q)=w(q
)-1
4203 GO SUB 8000: LET qw=q: LET
nr=nr1: GO SUB 1700: GO SUB 1200
: LET s(q,nr1)=pl: LET f(apl)=0:
 LET f(pl)=q: GO TO 4020
4900 PRINT AT 13,23;"Illegal";AT
 14,23;"move !": PAUSE 30: BEEP
.1,7: BEEP .1,5: BEEP .1,7: PAUS
E 111: PRINT AT 13,23;"         ";
AT 14,23;"         ": GO TO 4110
5000 REM Start
5010 PAPER 7: INK 1: GO SUB 1500
5500 FOR q=1 TO 4
5505 IF d(q)=4 THEN GO TO 5530
5510 IF q(q)=1 THEN GO SUB 4000
5520 IF q(q)=2 THEN GO SUB 6000
```

Scan: S. Höltgen

Bit-Parade

Nichts als Zahlen kennt der Computer oder genauer: nur die Zustände Aus (0) und Ein (1). Daß man trotzdem mit ihm rechnen kann, wird klar, wenn man eine Handvoll Fachchinesisch verstanden hat. Hier der zweite und letzte Teil der im vorigen HC-Heft begonnenen Bit-Parade.

Offset
Die 65536 Speicherzellen eines 64-KByte-Rechners sind fortlaufend durchnumeriert von 0 bis 65535. Die Anzahl Zellen von einer bestimmten Adresse bis zu einer bestimmten anderen nennt man Offset.
Beispiel Zeichensatz: Er umfaßt 256 Zeichen, von denen jedes einzeln 8 Byte belegt (s. Bit-Muster), zusammen sind das genau 2 KByte.
Die 256 Zeichen sind durchnumeriert von 0 bis 255, jedes Zeichen wird über diesen (ASCII-)Code aufgerufen, und das geht so:
Da jedes Zeichen acht Speicherzellen belegt, liegen vor dem Zeichen mit dem Code genau n ∗ 8 Adressen. Dieser Betrag heißt Offset.
Der Zeichensatz beginnt mit einer bestimmten Basis-Adresse. Die Daten des Zeichens mit dem Code n beginnen bei: Basisadresse + Offset (n ∗ 8).

Page
Der Speicher eines 64-KByte-Rechners besteht aus 65536 Speicherzellen. Um diese zu adressieren, werden zwei Byte benötigt: 256 ∗ 256 = 65536. Der Adreßraum gliedert sich dadurch in Blöcke von 256 Zellen, ein solcher Block wird Page (Seite) genannt. Das höhere Byte (high Byte) adressiert die Page, das niedrigere (low) die Zelle auf der Page.

Peek
Dieser BASIC-Befehl stellt eine Verbindung zum Innenleben des Computers her. PEEK liest den Inhalt einer bestimmten Adresse, doch die Ausgabe (mit PRINT) erfolgt in dezimaler Form.
Format: PEEK (Adresse)

POKE
Mit POKE kann eine RAM-Speicherzelle beschrieben werden. Manche Speicherbereiche (ROM) können nicht beschrieben werden, andere werden vom Betriebssystem laufend und so schnell überschrieben, daß eine Änderung aus dem Schnecken-BASIC praktisch wirkungslos bleibt.
Format: POKE Adresse,Dezimalwert

Vektor
Bei der Speicherverwaltung versteht man unter einem Vektor einen Zeiger auf eine bestimmte Adresse, z.B. die Basis-Adresse des Bildschirmspeichers.
Um die Nummer einer Adresse zu erfassen, werden zwei Byte, also auch zwei Speicherzellen benötigt. Der Dezimalwert zwischen 0 und 65535 (Nummer der angezeigten Adresse) wird in zwei Byte zerlegt. Die beiden Byte werden als LO(w)-Byte und HI(gh)-Byte bezeichnet und auch in dieser Reihenfolge gespeichert.
LO + HI ∗ 256 = gesuchte Adresse
Ist der Vektor auf den Bildschirmspeicher in den Adressen n und m abgelegt, dann findet BASIC die gesuchte Basis-Adresse durch:
PEEK(n) + PEEK(m) ∗ 256
In manchen Fällen wird auf das LO-Byte verzichtet. Der Vektor belegt dann nur ein Byte, ist mit 256 zu multiplizieren und kann nur auf Adressen am Anfang einer Page (s. dort) weisen.

Wort
Mit dem Begriff Wort bezeichnet man die Menge von Bit, die der Prozessor parallel verarbeiten kann. Bei einem Home-Computer, der wie üblich acht Bit schluckt, ist ein Wort ein Byte groß. Bei PC, die sechzehn Bit verschlingen, ist ein Wort entsprechend zwei Byte groß usw. *Karl-Heinz Koch*

Programm-Bibliothek

Wichtige Hilfsroutinen, auf die man immer wieder zurückgreifen kann (Teil 16)

30. Winkel zwischen Vektoren

Die nachfolgende Routine berechnet aus den eingegebenen Koordinaten zweier Vektoren deren Länge, den Winkel zwischen den Vektoren und die Größe des Skalarproduktes. Die Anzahl N der Koordinaten (Dimension) wird vorgegeben.

Der Aufruf

Dem Unterprogramm werden ab Zeile 1000 die Anzahl N und die entsprechende Zahl A(N) bzw. B(N) von Koordinaten der beiden Vektoren übergeben. Vor der Eingabe der Koordinaten erfolgt die Anlegung der entsprechenden Felder durch eine DIM-Anweisung. Das Programm berechnet in den Variablen Y, W bzw. Z, W1, W2 den Winkel zwischen den beiden Vektoren im Bogenmaß, Gradmaß mit Dezimalbruch bzw. Gradmaß in Minuten und Sekunden. Die Ausgabe der Werte erfolgt in den Zeilen 720 bis 810.

Die Routine

Die Berechnung des Winkels erfolgt über das Skalarprodukt. In der Schleife von 1010 bis 1030 werden hierzu die Quadrate der Längen der Vektoren und das Skalarprodukt berechnet. In Zeile 1035 wird die Cosinus-Funktion des Winkels ermittelt, deren Wert in Zeile 1060 in das Bogenwinkelmaß umgerechnet wird.

```
100 REM INPUT
120 PRINT CHR$(147)
140 CLR
200 REM EINGABEN
210 OPEN 1,0
220 PRINT " DIMENSION ";
230 INPUT# 1,N : DIM A(N), B(N)
240 PRINT
250 PRINT" 1.VEKTOR : ";
260 FOR I=1 TO N
270 INPUT# 1,A(I)
280 NEXT
290 PRINT
300 PRINT" 2.VEKTOR : ";
310 FOR I=1 TO N
320 INPUT# 1,B(I)
330 NEXT
340 CLOSE 1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT : PRINT
720 PRINT " SKALARPRODUKT"P
730 PRINT " LAENGE DES 1.VEKTORS ";
740 PRINT SQR(R)
750 PRINT " LAENGE DES 2.VEKTORS ";
760 PRINT SQR(S)
770 PRINT " COS(X) =";X
780 PRINT " WINKEL IN RAD ";Y
790 PRINT " WINKEL IN GRAD ";W
800 PRINT " WINKEL IN GRAD ";
810 PRINT Z;W1"'";W2"''"
820 PRINT : PRINT
830 GOTO 140
1000 REM
1005 P=0 : R=0 : S=0
1010 FOR I=1 TO N
1015 P=P+A(I) * B(I)
1020 R=R+A(I) * A(I)
1025 S=S+B(I) * B(I)
1030 NEXT
1035 X=P/SQR(S * R)
1040 IF R=S THEN X=P/R
1050 IF X=-1 THEN Y=π : GOTO 1065
1055 IF X=1 THEN Y=0 : GOTO 1065
1060 Y=π/2-ATN(X/SQR(1-X * X))
1065 W=(Y/π) * 180 : Z=INT(W)
1070 W1=(W-INT(W)) * 60
1075 W2=(W1-INT(W1)) * 60
1080 W1=INT(W1): W2=INT(W2+.5)
1085 RETURN
```

Scan: S. Höltgen

31. Vektorprodukt

Die nachfolgende Routine berechnet für N – 1 Vektoren der Dimension N den N-ten Vektor, der auf den vorgegebenen N – 1 Vektoren senkrecht steht, das heißt, das Skalarprodukt dieses so berechneten Vektors ist mit allen gegebenen Vektoren Null.

Der Aufruf

Dem Unterprogramm ab Zeile 1000 werden die Dimension N der Vektoren und die Anzahl der Koordinaten der entsprechenden Zahl von Vektoren übergeben. Vor der Eingabe der Koordinaten erfolgt eine Dimensionierung der notwendigen Felder. Die Ausgabe der Komponenten V(I) des Produktvektors erfolgt in Zeile 732.

Die Routine

Die Routine berechnet nach dem Determinantenverfahren (vgl. Ausgabe HC Mai 1986) die Koordinaten des Vektorproduktes V(I) innerhalb von vier geschachtelten Schleifen (Zeile 1010, 1100 bzw. 1105, 1115) Aus den Zwischenwerten werden in Zeile 1115 die einzelnen Komponenten V(Z) des Produktvektors ermittelt.

```
100 REM INPUT
110 PRINT CHR$(147)
120 CLR
130 OPEN 1,0 : PRINT
140 PRINT TAB(3)" DIMENSION : ";
150 INPUT#1,N : PRINT
160 DIM B(N,N),A(N,N),S(N),V(N),H(N)
170 PRINT : PRINT TAB(2);
180 FOR I=1 TO N-1
190 PRINT TAB(2)I". VEKTOR : ";
200 FOR J=1 TO N
210 INPUT#1,B(J,I)
220 NEXT J
230 PRINT : PRINT TAB(2) : NEXTI
240 CLOSE 1
500 GOSUB 1000
700 REM OUTPUT
710 PRINT : PRINT :
720 PRINT TAB(3)"ERGEBNISVEKTOR: "
730 PRINT : PRINT TAB(3) : FOR I=1 TO N+1
732 PRINT V(I); : NEXT : PRINT
733 PRINT
740 PRINT TAB(2)" NEUE WERTE?"
750 PRINT TAB(2)" TASTE J ODER N!"
760 G$=""
770 GET G$ : IF G$="J" GOTO 120
780 IF G$="N" THEN END
790 GOTO 760
1000 REM ROUTINE
1005 N=N-1
1010 FOR Z=1 TO N+1 : U=1
1015 IF Z/2=INT(Z/2) THEN U=-1
1020 FOR I=1 TO N
1025 FOR J=1 TO N
1030 A(I,J)=B(I+1,J)
1035 IF I<Z THEN A(I,J)=B(I,J)
1040 NEXT : NEXT
1045 REM KOMPONENTE
1050 M=N : D=1 : FOR K=1 TO M
1055 IF A(1,1)=0 THEN GOSUB 1200
1060 IF D=0 THEN K=N : GOTO 1105
1065 FOR I=M TO 1 STEP-1
1070 FOR J=2 TO M
1075 A(J,I)=A(J,I)-A(1,I)*A(J,1)/A(1,1)
1080 NEXT J : NEXT I
1085 S(K)=A(1,1) : M=M-1
1090 FOR I=1 TO M : FOR J=1 TO M
1095 A(I,J)=A(I+1,J+1)
1100 NEXT J : NEXT I
1105 NEXT K
1110 FOR I=1 TO N : D=D*S(I) : NEXT
1115 V(Z)=D*U : NEXT Z
1120 RETURN
1200 FOR L=1 TO M
1205 IF A(L,1)<>0 THEN 1215
1210 NEXT: D=0 : RETURN
1215 FOR P=1 TO M
1220 H(P)=A(1,P) : A(1,P)=-A(L,P)
1225 A(L,P)=H(P) : NEXT : RETURN
```

Die Programmbeispiele laufen ohne Änderungen auf dem Commodore-Home-Computer, für andere können Anpassungen nötig sein.

Betrifft Billard-Wettbewerb und Umschreiben von C64-Programmen auf C16

Aufgrund der Einstellung von HC können wir leider nicht mehr über das Ergebnis der beiden Wettbewerbe berichten. Sie erfahren alles Wissenswerte darüber, jedoch in einer späteren Ausgabe von CHIP beziehungsweise eventuell einem Heft aus der CHIP-Special-Reihe.

Irrtum bei der Auswertung des Primzahl-Wettbewerbs

Bedauerlicherweise ist der Redaktion bei der Ermittlung des Siegers des Primzahl-Wettbewerbs ein fataler Fehler unterlaufen. Es sollte die größte von einem Homecomputer bestimmbare Primzahl gefunden werden. Eine ganze Reihe von Lesern verfiel hierbei auf eine Lösungsmethode, die auch vom vermeintlichen Sieger, Herrn Pfefferle, angewandt wurde. Leider ist diese Methode falsch, wie die Redaktion (leider zu spät) einsehen mußte.

Unter anderem erreichte uns hierzu eine ausführliche Darlegung von Herrn Dr. Wilfried Keller vom Regionalen Rechenzentrum in Hamburg. Hier ein Auszug: Das Konkstruktionsprinzip der erzeugten Zahlen wurde bereits von Euklid verwandt, um zu beweisen, daß es unendlich viele Primzahlen gibt. Der Beweis nimmt an, daß es endlich viele Primzahlen gibt und führt dies zu einem Widerspruch. Wenn es nämlich endlich viele Primzahlen gibt, dann kann man sie wie folgt durchnumerieren:

$p_1, p_2, p_3, \ldots p_n$

Bildet man nun die Zahl

$Z = p_1 * p_2 * \ldots * p_n + 1$,

so erkennt man, daß Z durch keine der Primzahlen p_1 bis p_n teilbar ist, denn der Divisionsrest beträgt in jedem Fall Eins. Es gibt nun zwei Möglichkeiten: Entweder ist Z selbst eine Primzahl, oder Z ist teilbar durch eine Primzahl, die in der Liste $p_1, \ldots, p_n$ noch nicht enthalten ist. In jedem der beiden Fälle hat man jedoch eine weitere Primzahl gefunden.

Leider wird der zweite Fall oft unterschlagen. Dies ist falsch, denn es trifft nicht zu, daß

$p_1 * p_2 * \ldots * p_n + 1$

eine Primzahl sein muß. Hier nur ein Gegenbeispiel:

$2 * 3 * 5 * 7 * 11 * 13 + 1$
$= 59 * 509$

Herrn Pfefferle trifft keine Schuld an diesem Malheur, denn seine Teilnahme erfolgte sicher im guten Glauben an die Korrektheit seiner Methode. Da wir die Einsendungen mittlerweile alle wieder an die Absender zurückgeschickt haben, besteht nicht die Möglichkeit, den Wettbewerb neu auszuwerten. Wir bitten unsere Leser, dieses Mißgeschick zu entschuldigen.

Böse lächelnd beobachtet der Tyrann Mangar, wie seine bösen Gehilfen den Bürgermeister fertigmachen. Nun ist er endlich der Herrscher von Skara Brae. Doch da hat er sich total getäuscht! Denn in einem unterirdischen Labyrinth versammeln sich sechs Abenteurer, um mit Kraft und Magie Skara Brae zu befreien. Wer ihnen helfen will, den Tyrann aus den Schuhen zu hauen, sollte sich schleunigst auf die Socken machen. Wer wissen will, was wir außer Skara Brae noch zu bieten haben, dem schicken wir gern unseren Gesamtkatalog.

This collection is for all TI99/4A user.
If you like TI99 computer join at international forum present on AtariAge.com site.
Greetings from Italy. Astharot

ariolasoft
Von Experten für Experten.